TÉCNICO LISBOA

# Multi-Objective Bi-Level Optimization for Parameter Adjustment in Machine Learning

**Nuno Filipe Cortes Fernandes**

Thesis to obtain the Master of Science Degree in

**Mestrado Integrado em Engenharia Electrotécnica e de Computadores**

Supervisors: Prof. José Rui de Matos Figueira
Prof. Mário Alexandre Teles de Figueiredo

## Examination Committee

Chairperson: Prof. José Eduardo Charters Ribeiro da Cunha Sanguino
Supervisor: Prof. José Rui de Matos Figueira
Member of the Committee: Prof. Rui Fuentecilla Maia Ferreira Neves

**July 2021**

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Declaração

Declaro que o presente documento é um trabalho original da minha autoria e que cumpre todos os requisitos do Código de Conduta e Boas Práticas da Universidade de Lisboa.

# Acknowledgments

I would like to thank my supervisors Profs. José Rui Figueira and Mário Figueiredo for all their help and advice. I would also like to thank the Researcher Andreia Guerreiro for providing help with the hypervolume indicator algorithm. Lastly, I would like to thank University of Wisconsin Hospitals, Madison, namely Dr. William H. Wolberg for the breast cancer database.

# Abstract

In Machine Learning problems, classical approaches such as grid search are not viable methods for the computation of hyperparameters for higher dimension problems due to combinatorial explosion. The hyperparameter adjustment can be formulated as a bilevel optimization problem. While the lower-level optimizes the training stage parameters, the upper-level serves as the validation stage and optimizes the hyperparameters. These problems can also contain multiple objectives to optimize.

This work tests the proof of concept of a multi-objective bi-level optimization algorithm, in particular evolutionary-based algorithms, to solve multi-objective Support Vector Machine problems with an automatic selection of hyperparameters. The selected algorithm is the Hybrid Bi-Level Evolutionary Multi-Objective Optimization algorithm and, in total, six formulations based on soft margin and total margin formulations were tested.

The formulations with the best results had similar results to the traditional dual formulation Support Vector Machine. The formulations with the objective based on the total margin formulation were found preferable since they achieved better performance in all four datasets. However, the classification problems were found to impact the observations and conclusions of the upper-level objective space of the algorithm.

In conclusion, the concept can be a reliable alternative and a good competitor to the classical Support Vector Machine algorithms.

# Keywords

Hyperparameter Optimization; Multi-Objective Bi-Level Optimization; H-BLEMO; Multi-Objective Support Vector Machine.

# Resumo

Em problemas de Aprendizagem Automática, as abordagens clássicas como a pesquisa com base numa grelha não são métodos viáveis para a computação de hiperparâmetros para problemas de maior dimensão, devido à sua explosão combinatória. O ajuste do hiperparâmetros pode ser formulado como um problema de otimização de dois níveis. Enquanto o nível inferior otimiza os parâmetros da fase de treino, o nível superior serve como a fase de validação e otimiza o hiperparâmetros. Estes problemas também podem conter múltiplos objetivos para otimizar.

Este trabalho testa a prova de conceito de um algoritmo de otimização multiobjetivo binível , em particular com algoritmos evolucionários, para resolver problemas de Máquina de Vetores de Suporte com multiobjetivos com seleção automática de hiperparâmetros. O algoritmo selecionado é o algoritmo Hybrid Bi-Level Evolutionary Multi-Objective Optimization e, no total, seis formulações baseadas nas formulações soft margin e total margin foram testadas.

No geral, os resultados são semelhantes à formulação dupla de Máquina de Vetores de Suporte tradicional. As formulações com objetivo baseado na formulação total margin foram consideradas preferíveis, uma vez que obtiveram um melhor desempenho nos quatro conjuntos de dados. No entanto, os problemas de classificação têm um impacto nas observações e conclusões do espaço objetivo de nível superior do algoritmo.

Em conclusão, o conceito pode ser uma alternativa fiável e um bom concorrente aos algoritmos clássicos de Máquina de Vetores de Suporte.

# Palavras Chave

Otimização de Hiperparâmetros; Otimização Multiobjetivos Binível; H-BLEMO; Máquina de Vetores de Suporte com Multiobjetivos.

# Contents


**1 Introduction and Motivation** **2**

**2 Revision of Literature** **5**

2.1 Multi-Objective Machine Learning Problems . . . . . 6

2.2 Concepts of Optimization Problems . . . . . 11

2.2.1 Multi-Objective Optimization . . . . . 11

2.2.2 Bi-Level and Bi-Level Multi-Objective Optimization . . . . . 12

2.3 Classical and Evolutionary Bi-Level, Multi-Objective and Multi-Objective Bi-level Algorithms 14

2.4 Hyperparameter Bi-Level Approach . . . . . 17

**3 Methodology** **19**

3.1 Hybrid Bi-Level Evolutionary Multi-objective Algorithm (H-BLEMO) Algorithm . . . . . 20

3.2 Proposed Multi-Objective Bi-Level Support Vector Machine Problems . . . . . 22

3.2.1 Support Vector Machine Formulations . . . . . 22

3.2.2 Objectives for Support Vector Machine Evaluation . . . . . 23

**4 Classification Task and Results** **25**

4.1 Pre-Testing . . . . . 26

4.2 Implementation Details and Observations . . . . . 27

4.2.1 Behavior of Classical Multi-Objective Bi-Level Problems . . . . . 27

4.2.2 Upper-Level and Lower-Level Variable Vector Interval . . . . . 29

4.2.3 Upper-Level Objective Space . . . . . 33

4.2.4 Termination Criteria . . . . . 35

4.2.5 Local Search . . . . . 39

4.2.6 Feature Transformation . . . . . 41

4.3 Classification Results . . . . . 41

4.3.1 H-BLEMO Results . . . . . 42

4.3.2 Comparison with Dual Formulation Algorithm Results . . . . . 45

**5 Conclusions and Further Work** **47**

5.1 Conclusions . . . . . 48

5.2 Further Work . . . . . 48

**A Accuracy Results** **56**

# List of Figures


2.1 Illustration of misclassified data points and their respective slack distance. . . . . 8
2.2 Example of an illustration of dominated and non-dominated points and True Pareto Front. 12
2.3 Flowchart of the NSGA-II algorithm. . . . . 16
2.4 Bi-Level formulation of hyperparameter optimization . . . . . 18

4.1 Pareto-optimal fronts of upper-level and some representative lower level optimization tasks are shown [Deb and Sinha, 2010]. . . . . 27
4.2 Solution of final archive of H-BLEMO [Deb and Sinha, 2010]. . . . . 28
4.3 Example of solution of final archive of constructed H-BLEMO. . . . . 28
4.4 Basic datasets where each colour represents a different class. . . . . 29
4.5 Effect of lower-level variable interval [-1;1] in the final archive solutions. . . . . 30
4.6 Effect of lower-level variable interval [-10;10] in the final archive solutions. . . . . 31
4.7 Effect of lower-level variable interval [-100;100] in the final archive solutions. . . . . 32
4.8 Comparison of pareto front of archive solutions of formulation 1 for the first test of Section 4.2.5. . . . . 34
4.9 Comparison of the effect of termination criteria in the final archive solutions of the linearly separable dataset with formulation 1. . . . . 36
4.10 Comparison of the effect of termination criteria in the final archive solutions of the non-linearly separable dataset with formulation 1. . . . . 37
4.11 Comparison of the effect of termination criteria in the final archive solutions for the Noisy dataset with formulation 1. . . . . 38
4.12 Comparison of the effect of Local Search in the final archive solutions for the linearly separable dataset with formulation 1. . . . . 39
4.13 Comparison of the effect of Local Search in the final archive solutions for the non-linearly separable dataset with formulation 1. . . . . 40
4.14 Three axes views of the Haberman's Survival dataset . . . . . 45

# List of Tables

2.1 Examples of feature transformation functions in SVM. . . . . 8

4.1 Accuracy (%) and standard deviation of runs for each formulation for the *Iris Setosa* and *Iris Versicolour* dataset. . . . . 42
4.2 Accuracy (%) and standard deviation of runs with different generations for each formulation for the *Iris Versicolour* and *Iris Virginia* dataset. . . . . 43
4.3 Accuracy (%) and standard deviation of runs with different generations for each formulation for the Noisy dataset (see Section 4.2.4). . . . . 43
4.4 Accuracy (%) and standard deviation of runs with different generations for each formulation for the Haberman's Survival dataset. . . . . 44
4.5 Accuracy (%) and standard deviation of runs with different generations for each formulation for the Wisconsin Breast Cancer dataset. . . . . 44
4.6 Accuracy (%) and standard deviation of test with soft margin SVM in the dual problem formulation for each dataset. . . . . 46

A.1 Noisy Dataset: Accuracy (%) of archive solutions for each run, and average and standard deviation of all runs for 100 and 200 generations for original and gaussian transformation. 57
A.2 *Iris Versicolour* and *Iris Virginia Dataset*: Accuracy (%) of archive solutions for each run, and average and standard deviation of all runs for 100 and 200 generations for original and gaussian transformation. . . . . 58
A.3 Haberman's Survival Dataset: Accuracy (%) of archive solutions for each run, and average and standard deviation of all runs for 100 and 200 generations for original and gaussian transformation. . . . . 59
A.4 Wisconsin Breast Cancer Dataset: Accuracy (%) of archive solutions for each run, and average and standard deviation of all runs for 100 and 200 generations for original and gaussian transformation. . . . . 60

# Acronyms

| | |
|---|---|
| **BP** | Bi-Level Optimization Problem |
| **CD** | Crowding Distance |
| **CV** | Cross Validation |
| **EA** | Evolutionary Algorithm |
| **EMO** | Evolutionary Multi-Objective Optimization |
| **H-BLEMO** | Hybrid Bi-Level Evolutionary Multi-Objective Optimization |
| **HO** | Hyperparameter Optimization |
| **ML** | Machine Learning |
| **MOBP** | Multi-Objective Bi-Level Optimization Problem |
| **MOP** | Multi-Objective Optimization Problem |
| **ND** | Non-Dominated Rank |
| **NSGAII** | Elitist Non-Dominated Sorting Genetic Algorithm |
| **PF** | Pareto Front |
| **SVM** | Support Vector Machine |

# 1

# Introduction and Motivation

The goal of Machine Learning (ML) is the development of algorithms and techniques that create a model to predict information and make decisions. The learning is made by providing data and solving an optimization problem by finding the set of optimal parameters that minimize a predefined expected loss function [Claesen and Moor, 2015]. In the case of, for example, a simple linear regression problem, the model is composed of 2 parameters where the loss function is, typically, the mean square error function. The construction of a model by the algorithm requires a selection of hyperparameters. These variables control the characteristics of the algorithm in training the model and have a significant influence on its performance. For example, in soft margin Support Vector Machine (SVM) models, a selection of a hyperparameter regularization constant and hyperparameter of the kernel is required. In the case of artificial neural network, they are the numbers of hidden layers, the number of hidden units, and the learning rate of the gradient descent algorithm. Generally, they are set before the training stage and are user-defined, which are a non-optimal selection.

Since it first appeared, several approaches have been developed to solve this optimization problem. The so-called classical approach consists of an exhaustive search or brute force strategy such as the Cross Validation (CV) strategy by employing a grid search procedure. It suffers, however, of several adversities, the main one being the fact that the combinatorial nature of this strategy leads to a combinatorial explosion as the dimension (number of features) of the problem increases. There are also assumptions regarding the objective functions such as, for example, convexity or differentiability. Of course, successful attempts have been made to improve the algorithm and approaches. These limitations motivated researches to use alternative approaches, such as the popular meta-heuristics, due to their flexibility. Despite appearing in the mid-1980s, the Evolutionary Algorithm (EA) based meta-heuristics have, only over the last 20 years, been significantly used and studied [Sinha et al., 2018, Sloss and Gustafson, 2020].

A recent alternative to the classical approach was proposed in the article by [Bennett et al., 2006]. The CV Hyperparameter Optimization (HO) problem was defined as a Bi-Level Optimization Problem (BP). The problem contains two mathematical programming problems where one is nested into the other. The BP has therefore two distinct levels, the outside one called upper-level or leader, and the other called lower-level or follower. The solution to the lower-level corresponds to the constraint functions of the leader. In HO, the lower-level corresponds to the optimization problem of the training stage and the upper-level to the optimization problem of the validation stage. Since, as mentioned above, the hyperparameters are chosen before training, they are upper-level variables. As for the model parameters, they are lower-level variables. The corresponding solution to the lower-level problem is the optimal model parameter set of training.

Although each level is composed of single objective function stage, the BP can be extended to include several objective functions in both or single level. This new formulation is referred to as Multi-Objective

Bi-Level Optimization Problem (MOBP).

Multiple objectives are often considered and grouped together into the same optimization function. However, the inexistence of conflicts between two or more objects cannot be guaranteed. Using multi-objective bi-level EA based meta-heuristic and the selection of multiple objective functions in each layer is the motivation for this thesis. The main contribution of this thesis is the proof of concept of MOBPs in the adjustment of hyperparameters and parameters, and the effect of SVM formulations with different objectives.

This work is organized as follows. Chapter 2 introduces some SVM problems with multiple objective in Section 2.1. Section 2.2.1 presents the mathematical concepts of multi-objective, and in Section 2.2.2 the bi-level and multi-objective bi-level optimization. It is followed by a brief state of the art review on classical and EAs applied to Multi-Objective Optimization Problems (MOPs) and MOBPs in Section 2.3, and the formulation of hyperparameter selection in BP in Section 2.4. Chapter 3 is dedicated to the description of the selected algorithm in Section 3.1, and to the selected SVM formulations in Section 3.2.1. The Classification results are in Chapter 4 and a comparison with traditional SVM algorithms is presented. Chapter 5 finalizes this work with conclusions about the results and overall work, and ideas for future work.

# 2

# Revision of Literature

## Contents


2.1 Multi-Objective Machine Learning Problems . . . . . 6

2.2 Concepts of Optimization Problems . . . . . 11

2.3 Classical and Evolutionary Bi-Level, Multi-Objective and Multi-Objective Bi-level Algorithms . . . . . 14

2.4 Hyperparameter Bi-Level Approach . . . . . 17

In Chapter 1, the difficulty of determining the best set of hyperparameters and the parameters that lead to the best performance of a ML model for higher dimension was stated.

In this Chapter, the first Section will introduce several types and examples of classification problems with SVM. Section 2.2 will be dedicated to multi-objective and bi-level optimization problems, which will be formally defined and thier respective concepts described. The following Section will be concerned with existing MOP and MOBP, specifically the EAs. The Chapter will end with a short introduction of the BP approach to HO and ML problems.

## 2.1 Multi-Objective Machine Learning Problems

ML has the objective of creating algorithms and techniques of automatic learning for a given data to provide accurate estimations. The two more known type of approaches are supervised learning where classification or regression problems are solved, unsupervised learning where pattern searching problems are computed by a given non label data.

For the case of classification in supervised learning, the goal of algorithms is to build a mathematical model that evaluates the given data and returns a label. Generally, these algorithms are linked to optimization as most models require a minimization problem of a loss function and optimization of model parameters.

The model is divided into two steps, the selection of hyperparameters and the optimization of parameters. The typical hyperparameters include the number of layers and nodes in neural networks, or a regularization hyperparameter in the Lasso algorithm, or the SVM hyperparameters of the slack variable. These hyperparameters have a fundamental role in the algorithm. They are weights that guide the learning process of the parameters. Something relevant to point out is that they are, in most cases, defined before training. This makes it difficult task to select since, in most problems, no prior knowledge of optimal value is known.

The ML algorithms can contain several objectives to optimize, such as the maximization of a classification metric, or the minimization of training error and validation error, or minimization of the total number of support vectors in SVM.

These objectives can be together in the same function or the type of problem can demand or allow multiple objective in different functions. This work focuses on classification problems based on SVM optimization. SVM is a prediction model developed in the 1990s by [Cortes and Vapnik, 1995] for pattern recognition. Used in binary classification, it employs the determination of the optimal hyperplane that separates the two classes.

Considering two classes 1 and -1, the training data $(x_1,y_1),\ldots,(x_n,y_n)$, y $\in$ {-1,1} and $x \in \mathbb{R}^d$ (d is

the dimension of feature space) and the following linear function

$$f(x) = w^t x + w_0, \tag{2.1}$$

the classes of $x$ are defined by

$$\hat{y} = sign(w^t x + w_0). \tag{2.2}$$

The optimization of an SVM boils down to the determination of the norm of the hyperplane and the constant $w_0$ defined in at least one point of each class that divide the training data as

$$y_i(w^t x_i + w_0) \leq 1 \quad \forall i. \tag{2.3}$$

The two hyperplanes are compacted in one in equation 2.3. The points belonging to the equation are called support vectors and, when identified, the remaining points are unnecessary to the classification task. However, the real problem is finding which ones are the support vectors. To ensure the optimal separation, the distance of the support vectors must be maximized. This distance is the separating distance of the two hyperplanes, commonly known as the margin, and is equal to $\frac{2}{\|w\|_2}$.

With equation 2.3 and considering the minimization approach to the margin, the SVM Hard formulation is defined by

$$\begin{aligned} \min_{w,w_0} \quad & \frac{1}{2}\|w\|_2^2 \\ \text{subject to} \quad & y_i(w^T x_i + w_0) \geqq 1. \end{aligned} \tag{2.4}$$

The final result requires at least a pair of support vectors to define the hyperplane, one for each class, and only these points from all data are necessary to store. The optimization should be aiming to keep good performance and simultaneously contain a small set of support vectors. One disadvantage of a model with a large number of support vectors is the possibility of overfitting. On this account, a validation stage in conjunction with the cross-validation technique is required after the training stage to cancel the disadvantage.

In many real-world problems, the data is rarely perfectly separable and usually contains noise. To relax constraint 2.3 it is allow slightly misclassified data points through the usage of a positive slack variable $\xi$. Even with misclassification, better overall performance is achieved when compared to formulation 2.4. This variable measures the error of misclassified points and is defined as the distance points to their respective class hyperplane as shown in figure 2.1. The formulation is known as soft margin SVM and is given by

$$
\begin{aligned}
\min_{w,w_0} \quad & \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{l}\xi_i \\
\text{subject to} \quad & y_i(w^T x_i + w_0) \geqq 1 - \xi_i, \\
& \xi_i \geqq 0, i = 1, \ldots, l.
\end{aligned}
\tag{2.5}
$$

The parameter C is a non-negative regularisation variable that controls the importance of misclassified points. In other words, the significance given to the optimization of the margin decreases for higher values leading to a smaller margin.

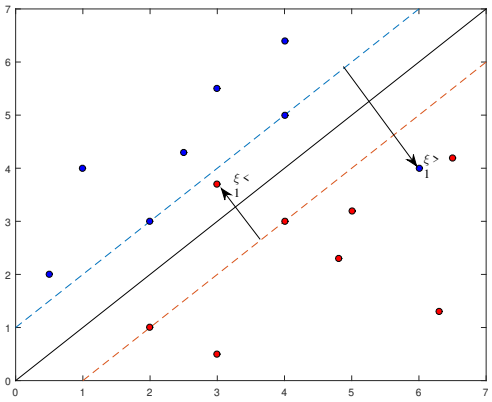



**Figure 2.1:** Illustration of misclassified data points and their respective slack distance.

For nonlinear problems, the introduction of a feature transformation is required and was motivated by an existing higher-dimensional space compared to the data where the mapping of data into the new space allows a linearly separation. This concept is significant as it generalizes the SVM formulation. In formulation 2.5 the points $x_i$ in the constraints are modified by a feature transformation function $z_i = \Phi(x_i)$. Some examples of used functions are shown in table 2.1. However, the utilization of a feature or kernel transformation is not a guaranteed improvement since these depend on a set of hyperparameters where the best varies for different datasets.

**Table 2.1:** Examples of feature transformation functions in SVM.

| Feature Transformation | Function | Hyperparameter |
|---|---|---|
| Linear | $Ax$ | $A$ |
| Polynomial | [ all monomials of degree up to p, with scaling depending on A]$^T$. Ex ($d = 2$): $[A, \sqrt{(2A)}x_1, \ldots, \sqrt{(2A)}x_d, x_1^2, , x_1x_2, \ldots, x_1x_d, \ldots, x_d^2]^T$ | $A, d$ |
| Gaussian | $exp\left(\frac{\|x\|_2^2}{2\sigma^2}\right)$ | $\sigma$ |

Generally, the formulated SVM problems, also named the primal problem, are optimized in a refor-

mulated problem called dual problem. Reformulating formulation 2.5 into a lagragian

$$L(w, w_0, \alpha) = \frac{1}{2}\|w\|_2^2 + \sum_{i=1}^{n} \alpha(1 - y_i(w^T x_i + w_0) - \xi_i) + C1^T\xi - \mu^T\xi \tag{2.6}$$

and minimizing in respect to $w$, $\xi$ and $w_0$

$$\begin{cases} \dfrac{\partial L(w, w_0, \xi, \alpha, \mu)}{\partial w_0} = \displaystyle\sum_{i=1}^{n} \alpha y_i = 0; \\ \nabla_w L(w, w_0, \xi, \alpha, \mu) = w - \displaystyle\sum_{i=1}^{n} \alpha y_i x_i = 0; \\ \nabla_\xi L(w, w_0, \xi, \alpha, \mu) = C - \alpha - \mu = 0, \end{cases} \tag{2.7}$$

the new formulation with kernel transformation is given by

$$\begin{aligned} \min_{w, w_0} \quad & \sum_{i=1}^{L} \alpha_i - \sum_{i=1}^{L} \alpha_i \alpha_j \Phi(x_i)\Phi(x_j) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C \\ & \sum_i \alpha_i y_i = 0 \end{aligned} \tag{2.8}$$

The dual formulation is particularly beneficial in nonlinear datasets and when using the kernel transformation. Since computing the mapping of the transformation in the primal formulation can be computationally expensive, using the kernel function $K(x, x) = \langle \Phi(x), \Phi(x') \rangle$ where $\Phi$: $\chi \to \mathcal{F}$ and $\mathcal{F}$ is a Hilbert Space, the computation is reduced to dot-product between points of transformed dataset in a total of N by N evaluations. Also, the fact that only the support vectors have $\alpha$ non-zero values facilitates the optimization and reduces the complexity.

The soft margin formulation had only in mind the distances of wrongly classified data points. The soft margin formulation had only in mind the wrongly classified data points. The distances of correctly classified data can also be taken into consideration. In other words, all points distances are taken into account. The idea was proposed in [Min Yoon et al., 2003], where the opposite concept of the slack variable, called surplus variable, $\xi^+$ or $\eta$, was introduced in the formulation 2.5. However, if for the slack case the objective was to minimize, in this case the distances are maximized. This soft margin SVM extension is referred to as Total Margin SVM and is expressed in 2.9. Two hyperparameter were introduced to control the trade-off of the slack vector and the surplus vector in respect to the margin minimization. The variable $C_1$ is selected to be higher than $C_2$, to ensure that at least one $\xi_i$ and $\eta_i$ are zero.

$$
\begin{aligned}
\min_{w,w_0} \quad & \frac{1}{2}\|w\|_2^2 + C_1 \sum_{i=1}^{l} \xi_i - C_2 \sum_{i=1}^{l} \eta_i \\
\text{subject to} \quad & y_i(w^T \Phi(x_i) + w_0) \geqq 1 - \xi_i + \eta_i, \\
& \xi_i \geqq 0, \eta_i \geqq 0, i = 1, \ldots, l,
\end{aligned}
\tag{2.9}
$$

Based on $\nu$-Support Vector Regression [Schölkopf et al., 1998] (modified version of $\varepsilon$-Support Vector Regression [Vapnik, 1995], where the optimization problem tries to determine a tube of predefined radius $\varepsilon$ that encapsulates the data), an equivalent formulation for classification was presented in [Schölokopf et al., 2000]. The formulation is named $\nu$-SVM and is defined as follows

$$
\begin{aligned}
\min_{w,w_0} \quad & \frac{1}{2}\|w\|_2^2 - \nu\rho + \frac{1}{l} \sum_{i=1}^{l} \xi_i \\
\text{subject to} \quad & y_i(w^T \Phi(x_i) + w_0) \geqq \rho - \xi_i, \\
& \rho \geqq 0, \xi_i \geqq 0, i = 1, \ldots, l, \\
& 0 \leqq \nu \leqq 1.
\end{aligned}
\tag{2.10}
$$

This design introduces a new objective into formulation 2.5 given by variable $\rho$ that corresponds to the minimum distance between all correctly classified points and the separating hyperplane control by the hyperparameter $\nu$. The hyperparameter C is equal to 1 since the authors concluded that, in the dual formulation, the variable did not affect the optimization problem. Similarly to slack and surplus variables, the opposite objective to the one presented before can be formulated by replacing the maximization of $\nu\rho$ with the minimization of $\mu\sigma$ where $\sigma$ represents the maximum distance between all misclassified points and the hyperplane.

These two objectives can be joined together identically to formulation 2.5 and form $\mu$-$\nu$-SVM formulation

$$
\begin{aligned}
\min_{w,w_0} \quad & \frac{1}{2}\|w\|_2^2 + \mu\sigma - \nu\rho \\
\text{subject to} \quad & y_i(w^T \Phi(x_i) + w_0) \geqq \nu - \sigma \\
& \sigma \geqq 0, \rho \geqq 0, i = 1, \ldots, l, \\
& 0 \leqq \mu, \nu \leqq 1.
\end{aligned}
\tag{2.11}
$$

So far, no specific information about the training data, except for noise, was mentioned. In real-world problems, the training sets commonly contain several obstacles such as unbalanced data or outliers. The presence of one hyperparameter gives equal importance to both classes. This assumption makes

an inappropriate approach to unbalanced training data cases. To overcome this issue, separable slack hyperparameters C were applied to formulation 2.5 in [Morik et al., 1999] for each class. Other approach was made to formulation 2.10 in [Hong-Gunn Chew et al., 2001] by reintroducing the hyperparameter. In the case of outliers, similar formulation to the 2.5 was presented in [Yoon et al., 2003], where an upper limit of the variable $\xi_i$ was limited by an upper bound, $\xi_{max}$, which proves itself especially functional to data sets .

## 2.2 Concepts of Optimization Problems

### 2.2.1 Multi-Objective Optimization

As in the case of non linearly separable SVM formulation, problems can contain multiple objectives in a single function. For some, the combination and the actual optimization of various objectives into a single function can be a complex task. In MOP, a parallel optimization of multiple functions is performed.

Considering the objective function $F : \mathbb{R}^n \to \mathbb{R}$, and the constraints $G_k : \mathbb{R}^n \to \mathbb{R}$, k = 1,. . . , K and $H_p : \mathbb{R}^n \to \mathbb{R}$, p = 1,. . . , P, the MOP is given by

$$\begin{aligned} \min_{x \in X} \quad & F(x) = (F_1(x), \ldots, F_t(x)) \\ \text{subject to} \quad & G_k(x) \leq 0, k = 1, \ldots, K \\ & H_p(x) = 0, p = 1, \ldots, P. \end{aligned} \tag{2.12}$$

Often, contradicting objectives exist, which creates conflicts between them. As a result, the optimization achieves the best result by finding the solution set where all functions are optimized. For a known solution set, this means that, for all objectives, a better result might exist. However, a worse outcome for one or more objectives also exists. The existence of this set is the main difference in optimizing single or multiple functions. The following two definitions are required to define the solution set.

**Definition 2.2.1** (Dominance)**.** Given two vector $x$, $y \in \mathbb{R}^k$ , $x \leq y$ if $x_i \leq y_i$ for $i$ = 1, . . . ,$k$, and that $x \prec y$ ($x$ *dominates* $y$) if $x \leq y$ and $x \neq y$.

**Definition 2.2.2** (Non-dominated)**.** A variable vector $x \in \mathbb{X}$ is non-dominated with respect to $\mathbb{X}$ if there does no exist $x' \in \mathbb{X}$ such that $f(x') \prec f(x)$.

A point is then considered best or non-dominated if is best in one and not worst in all the other objectives. In figure 2.2, several examples of non-dominated points are depicted such as point 1 (1 dominates 6 in respect to $F_1$) and point 5 (5 dominates 9 in respect to $F_2$). In the decision variable space, the vector containing the points 1 to 5 is called efficient solution or Pareto optimal solution. As for the objective space, the same vector for the same points is referred to as Pareto Front (PF).

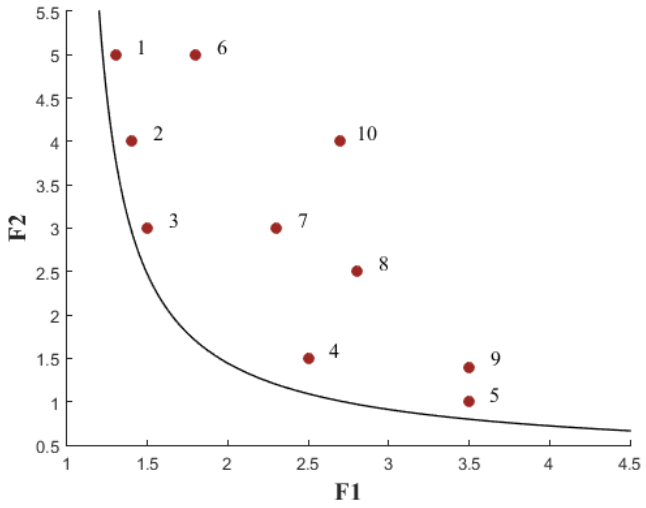



**Figure 2.2:** Example of an illustration of dominated and non-dominated points and True Pareto Front.

Although the existence of a larger number of function in the optimization problem implicates a higher probability of existing conflicts, this is not always the case. It might be natural to assume that adding more objective would turn the problem into a harder one. Except for complex models where adding more functions increases the difficulty of the problem, it was concluded in [Schutze et al., 2011] that the addition of many objective functions per se does not necessarily imply an increased hardness of the problem.

### 2.2.2 Bi-Level and Bi-Level Multi-Objective Optimization

In single-objective or MOP optimization, the constraints are assumed to be known. However, the set-valued mapping, that is, the set of constraints, might be itself defined as an optimization problem. This lead to the conceptualization of a constraint optimization and formulation of the BP.

The BP is a mathematical program composed of two levels of optimization. One is referred to as the leader or upper-level, and the other is the follower or lower-level. The names are indicative of their location in the bi-level framework. The upper-level is the main optimization problem, and the lower-level is the secondary optimization problem which is nested in the first one. The levels are characterized by their one objective function, constraints, and the class of decision vector variables. While the lower-level is optimized with respect to the lower-level decision vector, the upper-level decision vector act as a parameter. This implies a constraining nature of the lower-level concerning the upper-level. The nested characteristic is the primary source of difficulty in the optimization as it might introduce discontinuities and non-convexity. The designations leader and follower can be traced back to the 1950s in game theory today known as Stackelberg games. Later, they were reformulated into a mathematical formulation as

previously described.

The recent increase in studies of BP in the last 20 years can be explained by the wide variety of applications in real-life problems and the difficulty of solving non-simple problems. Applications can be found in Electricity Transmission (electric power, smart grid, distribution), Network design (traffic, transportation), Telecommunications (wireless, telecommunication interference), Supply Chain (supply, manufacturer, management inventory), ML (parameter adjustment) among others. For specific examples, refer to the compilation in [Sinha et al., 2018].

Considering formulation in 2.12 in both levels, the BP can be extended and transformed into MOBP. For the upper level objective function $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$ and the lower level objective function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^q$, the MOBP is defined by

$$\begin{aligned} &\underset{x_u \in X_U, x_l \in X_L}{\text{“min”}} && F(x_u, x_l) = (F_1(x_u, x_l), \ldots, F_p(x_u, x_l)) \\ &\text{subject to} && x_l \in \underset{x_l \in X_L}{\text{argmin}} \{f(x_u, x_l) = (f_1(x_u, x_l), \ldots \\ & && \qquad \ldots, f_q(x_u, x_l)) : g_j(x_u, x_l) \leq 0, j = 1, \ldots, J\} \\ & && G_k(x_u, x_l) \leq 0, k = 1, \ldots, K \end{aligned} \tag{2.13}$$

where $G_k : X_U \times X_L \to \mathbb{R}$, k = 1,..., K and $g_j : X_U \times X_L \to \mathbb{R}$ represent the upper level constraints and the lower level constraints, respectively. Both constraints can also have equality constraints. The quotation marks in 2.13 highlight the undefined decision of the upper level, due to the objective functions, of the lower level optimal solution. Since the upper-level optimizes in respect to the $x_L$ set and the lower-level solution is, in general, a set of optimal solutions, the best choice from this set is not clearly defined. The objective of a MOBP is the determination of the upper-level Pareto optimal solutions that approach to the theoretical PF (true PF) or at least an approximate PF. Due to this ambiguity, as indicated in [Alves and Antunes, 2017], four extreme attitudes or solutions were identified: optimistic solution, pessimistic solution, deceiving solution, and finally rewarding solution.

Unlike BP and as mentioned in 2.2.1, multi-objective optimization can lead to conflicts between objective functions, and there might not always exist a feasible solution that optimizes the MOBP. A feasible set of solutions to the upper-level is required to replace the concept of the optimal solution in BP to tackle this problem. The Pareto optimal solutions of the lower-level are the only set feasible to the upper-level optimization problem. The MOBP is solved by fixing an upper-level variable and optimizing the lower level for the lower level variable.

## 2.3 Classical and Evolutionary Bi-Level, Multi-Objective and Multi-Objective Bi-level Algorithms

The BPs are arduous to solve due to their nested structure. That makes the optimization of MOBPs more complex.

The most basic approach is the transformation of the two-level problem into a simple one-level optimization problem. By applying the Karush-Kuhn-Tucker conditions, the lower-level problem is transformed into a set of constraints to the upper-level, for example in [Dempe et al., 2006, Al-Khayyal et al., 1992]. Another approach is using the descend method with again a level reduction and applying the gradient on the upper-level function [Vicente et al., 1994] or Penalty functions methods [Aiyoshi and Shimizu, 1981, Shimizu and Aiyoshi, 1981] for nonlinear problems where the lower-level problem is substituted by a single or double [Ishizuka and Aiyoshi, 1992] penalty function. So, instead of solving the problem itself, a less complex problem is optimized, and for this reason, an approximated solution will be achieved. For some real-world problems this approximated solution is good enough. But in more complex ones with characteristics such as non-convexity, non-linearity, non-differentiability, classical algorithms fail or can not solve the problem.

The alternative to this are the EAs, in specific the sub-type named genetic algorithm. Based on the theory of survival of the fittest alongside genetic concepts such as mutation and recombination, it has the objective of applying these concepts to a population of members and iteratively creating new members, called offsprings or child solutions, to obtain better solutions compared with the previous population and generations. These algorithms were also later extended to problems containing multiple objective functions.

Pioneer by the author of VEGA algorithm [Schaffer, 1985] in the 1980s, the application of Evolutionary Multi-Objective Optimization (EMO) has been widely used and improved since then. Contrary to classical techniques that require several separate runs to compute the PF, as stated in [Coello, 1999], the EMOs are ideal for MOPs since a set of possible PF solutions in parallel is computed in a single run as well as being less susceptible to shape and continuity.

Over the years, several techniques and approaches were invented, and, by far, the most popular are the PF based approaches. One of the most used, tested, and widely established EMO is the improved version of the algorithm by the same authors in [Srinivas and Deb, 1994] and is referred to as Elitist Non-Dominated Sorting Genetic Algorithm (NSGAII) [Deb et al., 2002]. The main improvement is the introduction of elitism in the algorithm. With this concept, the previous parent population members can be contained in the child population, allowing the prevention of loss of good solutions and helping an overall better convergence [Zitzler et al., 2000]. It is important to mention that, as said in [Deb, 2010], the EMO are heuristic-based and, for this reason, do not guarantee convergence of exact PF as theoretically

supposed for tractable, such as linear or convex problems.

The main operators of changing or diversify the population of a genetic algorithm are the crossover, and mutation with a previous selection. The selection operator is the method where members of a population are selected by ranking the population with a fitness value. The principal fitness measure is the Non-Dominated Rank (ND) based on definition 2.2.2. Another important aspect of solutions in the PF is the requirement of diversity to ensure the complete representation of the PF. For this reason, and to help differentiate solutions with equal ND, the Crowding Distance (CD) measure is used. Measuring the density outside of the point by computing the cuboid form with the nearest neighbours as vertices when two solutions have the same ND rank, a solution with a bigger cuboid or less crowded region is preferable. This operator is called tournament selection.

The crossover or recombination is, as the name suggests, a reconfiguration of the parents' solutions to obtain new child solutions similar to the process of chromosomal recombination in biology. The NSGAII uses the SBX operator [Deb and Agrawal, 1995].

After the recombination, the mutation operator helps in diversifying the child solutions and preventing local minima by slightly changing the solutions. The NSGAII uses four parameters to control the progression of the operators: crossover probability, index for SBX operator, mutation probability, and the index of polynomial mutation. These have a crucial impact on the performance of the algorithm and vary on the optimization problem. To take into account this uncertainty of the standard parameters (crossover probability of 0.9, index for SBX operator of 15, mutation probability of 0.1, index of polynomial mutation of 20), in [Andersson et al., 2015, Andersson et al., 2016], the authors used MOP and BP approaches to tune these parameters using the Hypervolume Indicator to evaluate the performance. This is similar to the HO in BP presented in Section 2.4.

The overall procedure of NSGAII is represented in figure 2.3. Using a parent population of members $P_t$ of size $N$, the NSGAII for each generation creates another population $Q_t$ called child population with the above operators. When the total number of new creation is equal to the parent population, the two equal size populations are combined in a new population $R_t$. This population is used to create the new parent population by removing half the members. Ranking and sorting $R_t$ with ND and in turn CD, the worst solutions of size $N$ are rejected. The previous steps are repeated until the predefined maximum generation is achieved.

Another different MOP approach is the Multi-Objective Evolutionary Algorithm based on Decomposition [Zhang and Li, 2007]. As the last name suggests, the main problem is decomposed or transformed into a series of many new problems, called subproblems, that are optimized in parallel. This series is a set of single objective optimizations, and the combination of each solution represents the non-dominated vector solution of the original problem. The presence of many-objective problems was found to compromise the optimization, especially when using the CD calculation [Kukkonen and Deb, 2006] for more

than three objectives in PF based algorithms. However, the this algorithm did not suffer from this difficulty and was found to surpass the performance of NSGAII [Zhang and Li, 2007]. A general review of the framework and recent developments can be found in [Santiago et al., 2014].

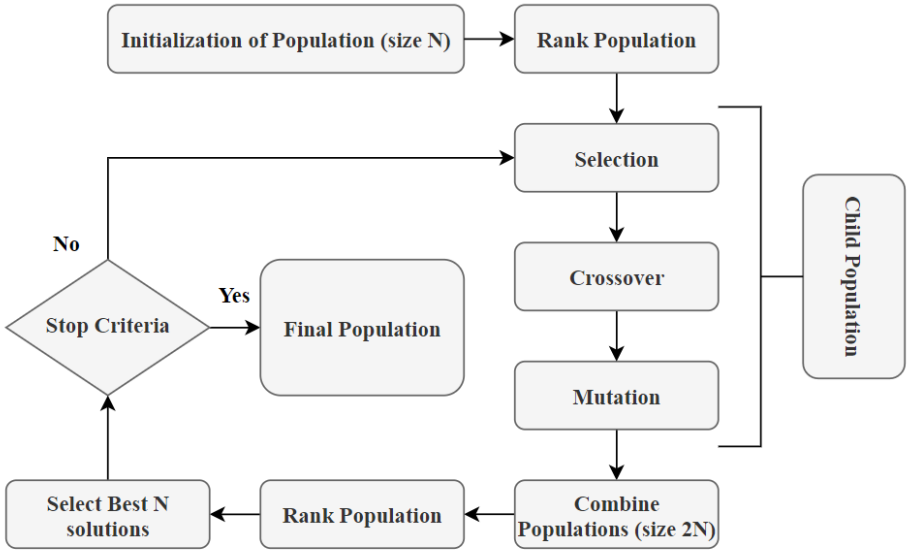



**Figure 2.3:** Flowchart of the NSGA-II algorithm.

Another approach is based on particle swarm optimization and is called Multi-Objective Particle Swarm Optimization [Coello Coello and Lechuga, 2002]. The algorithm uses a population or a swarm of solutions that iteratively update their position through the search space to improve their status to achieve the PF. They are referred to as a swarm since they emulate a flock of birds or shoal of fish, and the movement is control by a function of position, velocity, and the values of the distance of best position and current position of a particle, and distance of the global best position and the current position of the particle. Although not surpassing NSGAII, it shows the algorithm as a solid competitor. For a survey of variants of the Multi-Objective Particle Swarm Optimization approach and applications, see [Lalwani et al., 2013].

The difficulty of many-objective problems presented above and increased demand for better algorithm in the last decade lead to the improvement of the NSGAII algorithm with influence of Multi-Objective Evolutionary Algorithm based on Decomposition by adopting decomposition and reference points, originating the third iteration named Reference-Point based Many-Objective NSGA-II [Deb and Jain, 2014,Jain and Deb, 2014]. When testing problems with up to 15 objectives, the algorithm achieved good convergence, except for the higher dimension cases where it struggled with diversity and convergence to the PF. Although this algorithm was proposed to substitute NSGAII in many objective problems, a study was conducted by the authors in [Ishibuchi et al., 2016] to compare the performance of both al-

gorithms. They concluded that NSGAII is not necessarily worse. It depends significantly on the type of problem rather than just the quantity of functions.

It is important to mention the above algorithms, since most MOBP algorithm used in each level a MOP algorithm. One of the original authors of NSGAII also developed an approach for MOBP called Bi-Level Evolutionary Multi-Objective Optimization algorithm [Deb and Sinha, 2009]. Although the described procedure uses the previous EMO to solve both levels of optimization, as indicated by the authors, any other developed algorithm can be used. For an algorithm with a similar scheme and based on particle swarm optimization, see [Carrasqueira et al., 2015].

This algorithm was later acknowledged to contain several drawbacks leading therefore to a new extended version named Hybrid Bi-Level Evolutionary Multi-Objective Optimization (H-BLEMO) algorithm [Deb and Sinha, 2010]. In Section 3.1 of the following Chapter, the drawbacks and the description of the H-BLEMO will be presented.

## 2.4 Hyperparameter Bi-Level Approach

Traditionally, hyperparameters in ML are determined by a series of trial and choosing, in the end, the set of values that achieved the best performance. This is done by an exhaustive n-dimension grid search. Typically, a CV technique is combined with the grid to improve validation performance and avoid overfitting models. Also called brute force, this approach has the downside of a combinatorial explosion, causing it to be unreliable in problems of dimension higher than two [Bergstra and Bengio, 2012] which can reach up to hundreds [Bergstra et al., 2013].

Random search is another technique to improve the previous strategy and consists of randomly initializing hyperparameters and iteratively modifying their values to better ones, according to the minimization of a loss function. It was also shown in [Bergstra and Bengio, 2012] the improvement of random search over grid search, stating the primary reason to be the fact that not all the hyperparameters are equally essential to adjust.

Another technique is the Gradient Descent and was first used by [Bengio, 2000] for computing the gradient with respect to the kernel and the error. Although highly efficient when applicable, as mentioned in [Igel, 2005], the differential characteristic of the algorithm restrict the use of non-differential kernels and objective functions, such as the number of support vectors. This algorithm was also used alongside the BP in [Sinha et al., 2020].

To remove the problems of using the CV technique for a higher number of hyperparameters, [Bennett et al., 2006] proposed a new program of bilevel CV and tested on support vector regression model. This way, for each fold of the CV an automatic selector was included with the training of the model with the respective set of hyperparameters in the lower-level and the validation in the upper-level. This

approach is presented in 2.4. This was followed by numerous new approaches such as in [Kunapuli et al., 2012, Moore et al., 2011, Fischer et al., 2015] using reduction methods with KKT conditions, gradient descent, and locally replacing the lower level problem with its unique solution, respectively.

For more complex problems, some algorithms were created. For instance, to solve optimization problems with non-smoothness and non-convexity introduced by the $l_p$ regularizers, using scaled bilevel KKT conditions a new iterative algorithm was created by [Okuno et al., 2018].

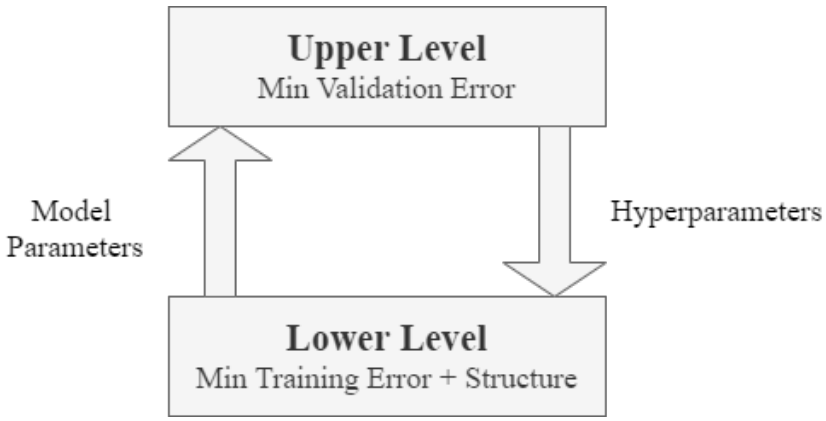



**Figure 2.4:** Bi-Level formulation of hyperparameter optimization

The same authors of the NSGAII algorithm also created their version of automated parameter tuning in [Sinha et al., 2014]. Instead of solving the nested problem, the lower-level optimization problem was replaced for the set-valued mapping of the lower-level by a quadratic approximation of the lower-level function. They also defined for the first time the mathematical formulation of the parameter tuning optimization problem.

The above approaches and algorithms can only deal with single objectives or functions with multiples objectives. As mention above, in most cases, no knowledge of the different objectives in the same function is known to have conflicts or benefits. Some authors created MOP approaches dealing separately with different objectives such as the number of support vectors and minimization of the classification error in [da Silva Santos et al., 2021] or in [Igel, 2005].

To the best knowledge found in this work, no attempt was done to incorporate both the multiple objectives functions and the bi-level formulation of figure 2.4. For this reason, this work focus on the application of MOBP evolutionary-based for an automated adjustment of the hyperparameters, and training and validation of SVM model with different formulations for classification problems.

# 3

# Methodology

## Contents

3.1 Hybrid Bi-Level Evolutionary Multi-objective Algorithm (H-BLEMO) Algorithm . . . . 20

3.2 Proposed Multi-Objective Bi-Level Support Vector Machine Problems . . . . . 22

In Chapter 2, the basic concepts of MOBP were presented in order to understand how to apply to ML problems. Also, several of these problems were given focusing mainly on the SVM model. This Chapter is dedicated to the description and explanation of the selected algorithm applied in Section 3.1 as well as the different SVM formulations in Section 3.2.1.

## 3.1 Hybrid Bi-Level Evolutionary Multi-objective Algorithm (H-BLEMO) Algorithm

As mentioned in Chapter 2, the H-BLEMO algorithm was the result of several improvements to the former algorithm.

First, the algorithm lacked sensibility in terms of the importance of solutions. Even for good or non-dominated vector solutions, the algorithm computes in the lower-level a user-defined number of solutions. This procedure is unnecessary for optimization and only worsens the computation time. For this reason, a self-adaptive process was created to select the number of lower-level solutions. By taking into account the ratio of the Euclidean distance of the new upper-level vector and the closest solution of the archive members, and the maximum Euclidean distance of all members in the archive, the size of the lower-level population was computed by the integer value of the multiplication of size and ratio. In this method, the farther the new vector is, the higher number of solutions is allocated to help the optimization.

Second, this lack of importance is also present in the maximum number of generations of the lower-level. Independently of the solutions, a user-defined number forced lower-level NSGAII are computed even for already found good solutions, leading to useless evaluations.

Third, for the termination of both levels, the algorithm made use of a user-defined criterion (the number of generation). To substitute it for a more sensitive method to the solutions, the $H_l$ metric is used and computed as follows:

$$H_l = \frac{H_l^{max} - H_l^{min}}{H_l^{max} + H_l^{min}}, \tag{3.1}$$

where maximum and minimum Hypervolume Indicator, respectively, are calculated for every $\tau$ generation (value used: 10).

Fourth, and most importantly, a local search operator is applied to lower-level solutions to ensure the requirement of solutions being lower-level optimal solutions and final archive solutions to be the true lower-level Pareto-optimal.

The summary of the procedure of H-BLEMO of a single generation is described below.

The main structure of the algorithm is composed of $n_s$ subpopulations. Each one shares the same upper-level variable vector, and, in total, the subpopulations have $N_u$ members. For each subpopulation,

a lower-level NSGAII is performed followed by a local search optimization. In every generation, the archive is updated after every lower-level optimization.

At the beginning of a generation, every member in the population $P_t$ of size $N_u$ and archive have computed the corresponding values of ND and CD for both levels. Step 1 deals with creating a new upper-level vector and respective lower-level vectors for the current generation. For a single upper-level vector creation, a binary tournament selection is applied to population $P_t$ and archive. Of the four outcome parents, two are selected stochastically are recombined using the SBX operator. Finally, one is mutated with the polynomial operator. This final vector is the new child upper-level vector. The aforementioned vector is then used to create $N_l$ child solutions, number which is based on its location in the current archive members' space.

After all child solutions are created for the upper-level vector, Step 2 performs the NSGAII to the lower-level. The algorithm differs solely from the original on the selection. Taking advantage of previous found archive solutions, if the subpopulation is present in the archive, only these are used in the binary tournament selection. Otherwise, the normal process is used. At the end of lower-level optimization, the solutions are sorted and ranked by ND and CD.

Step 3 involves the new optimization addition of the Local Search operator to achieve the locally PF. Since the operator can be expensive, as later verified to represent 50% of all computation effort in [Deb and Sinha, 2010], the operator was only applied to solutions that follow certain properties to exclude inadequate solutions. The operator is defined by the optimization of achievement scalarizing function problem [Wierzbicki, 1980].

Step 4 is for updating the archive after the Local Search. For only the deemed optimal solutions, these are compared with all archive members. If the solutions are non-dominated, these enter the archive, and the dominated members are excluded. In case of exceeding the maximum size of the archive, until the size is reached, the members are removed according to CD.

In Step 5 the creation of all new solutions finalizes, meaning the above steps are repeated until the population of new solutions has the exact size of the parent population $P_t$. What follows is the combination of both populations after a ranking by ND and CD for future selection of $N_u$ members.

This selection of half of the combined population is step 6 of the algorithm. The members first considered are those that have upper-level ND equal to 1, and then lower-level ND equal to 1 in order of reducing by lower-level CD. If the entire lower-level subpopulation is already present in the side population and the future solutions are from the same subpopulations and have both ND equal to 1, no further copy to the side population is done. The process is repeated for all upper-level ND equals 1 and future values until members reach $N_u$ size.

In the last step, for each subpopulation in the side population not created on the above steps, a lower-level NSGAII is utilized for helping the individual approximation of PF. The termination criteria

metric 3.1 is computed, and if the value reaches lower than the threshold on generations multiple of $\tau$, the algorithm comes to an end. Otherwise, the steps above are repeated.

## 3.2 Proposed Multi-Objective Bi-Level Support Vector Machine Problems

The previous algorithm was used to test two selected ML formulations from the aforementioned in Chapter 2, the formulation 2.5 and 2.9. These problems contain multiple objectives in their function. However, in most real problems the data might lead to conflicts between objectives which if occurred a worst performance is achieved. Depending on the problem, these conflicts might disappear if objectives are transformed into their one function.

### 3.2.1 Support Vector Machine Formulations

The two objectives in the selected primal formulations are the minimization of slack variable and maximization of surplus variable, and in total, six different formulations were created. The constraints remain the same as in the original formulation and for that reason are not shown below. The first two formulas were based on the formulation 2.5 and were defined as

Formulation 1:

$$\min_{w,w_0} \left\{ F_1 = \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{l}\xi_i \right. \tag{3.2}$$

Formulation 2:

$$\min_{w,w_0} \begin{cases} F_1 = \|w\|_2^2 \\ F_2 = \sum_{i=1}^{l}\xi_i \end{cases} \tag{3.3}$$

while the remaining were originated from the formulation 2.9 and were defined as

Formulation 3:

$$\min_{w,w_0} \left\{ F_1 = \frac{1}{2}\|w\|_2^2 + C_1\sum_{i=1}^{l}\xi_i - C_2\sum_{i=1}^{l}\eta_i \right. \tag{3.4}$$

Formulation 4:

$$\min_{w,w_0} \begin{cases} F_1 = \|w\|_2^2 \\ F_2 = C_1\sum_{i=1}^{l}\xi_i - C_2\sum_{i=1}^{l}\eta_i \end{cases} \tag{3.5}$$

Formulation 5:

$$\min_{w,w_0} \begin{cases} F_1 = \|w\|_2^2 \\ F_2 = \sum_{i=1}^{l} \xi_i \\ F_3 = -\sum_{i=1}^{l} \eta_i \end{cases} \tag{3.6}$$

Formulation 6:

$$\min_{w,w_0} \begin{cases} F_1 = \frac{1}{2}\|w\|_2^2 - C\sum_{i=1}^{l} \eta_i \\ F_2 = \sum_{i=1}^{l} \xi_i \end{cases} \tag{3.7}$$

### 3.2.2 Objectives for Support Vector Machine Evaluation

For evaluating the lower-level solutions in the training and validation stage, two types of objectives were selected. The first corresponds to the error of a hyperplane of the dataset and the other to the classification task.

The hinge loss is the most commonly loss function use in training SVM. It's defined for a particular point as

$$L_{Hinge}(y, w^T x + w_0) = max\{0, y(w^T x + w_0)\}, \tag{3.8}$$

where y is the output $\pm$ 1. This is an equivalent definition to the slack variable presented in Section 2.1.

For the classification of classes, the metric used is a specific case of the F-score [van Rijsbergen, 1979] defined as

$$\text{F-score} = (1+\beta^2)\frac{PrecisionRecall}{\beta^2 Precision + Recall}, \tag{3.9}$$

where Precision is the ratio of true positive labels in all predicted data as positive, Recall is the ratio of true positive labels in all correctly classified data, and $\beta$ the importance of the Precision over Recall. The metric is called F1-score because the commonly used value of $\beta$ is 1, meaning the same importance for the precision and recall. To transform this metric into a minimization problem the same approach in [Musicant et al., 2003] was applied. The author created a SVM formulation integrating the maximization of the F1-score fused with the slack variable. The new metric is given by

$$\text{F1-score} = \frac{1}{1 + \frac{1-C}{2z}}, \tag{3.10}$$

where C represents the global performance using the accuracy and z the ratio of true positive classifications. The maximization of the F1-score is achieved by the minimization of fraction in the denominator, assuming z $\neq$ 0, and that results in the following minimization problem

$$(1 - C) - 2z. \tag{3.11}$$

The main criticism on the utilization of the metric to validate the model lies in the fact that only the true positive cases are evaluated and can be misleading in unbalanced datasets. For this reason, the F1-score for negative cases was used. A similar approach to 3.11 can be formulated for the negative F1-score with a new z variable, $z_{neg}$, representing the ratio of true negative classifications.

In conclusion, in both levels and in all formulations, the hinge loss, the F1-score, and the negative F1-score were employed.

# 4

# Classification Task and Results

## Contents


4.1 Pre-Testing . . . . . 26

4.2 Implementation Details and Observations . . . . . 27

4.3 Classification Results . . . . . 41

In this Chapter, the results for the proof of concept of MOBP in ML are shown. In Section 4.1, the source of NSGAII and other algorithms used to adapted into the creation of the H-BLEMO as well as the basic parameters and the dataset used are indicated. This Section is followed by the decisions and commentary on the implementation of the algorithm, in specific on the behaviour of traditional MOBP in Section 4.2.1, on the selection of upper-level and lower-level interval in Section 4.2.1, on the upper-level objective space in Section 4.2.3, on the termination criteria in Section 4.2.4, on the Local Search operator in Section 4.2.5, and on the transformation of dataset feature in Section 4.2.6. Finally, the results of the various SVM formulation of each dataset are presented in Section 4.3.

## 4.1 Pre-Testing

The H-BLEMO constructed to this work is based on the Matlab tool Evolutionary multi-objective optimization platform PlatEMO [Tian et al., 2017] made available by BIMK Group, specifically the multi-objective algorithm NSGAII. As for the termination criteria, the hypervolume indicator algorithm used was proposed by [Fonseca et al., 2006] version 1.3 and can be found in [Fonseca et al., 2017]. For both levels, the standard parameters of NSGAII (crossover probability of 0.9, index for SBX operator of 15, mutation probability of 0.1, index of polynomial mutation of 20) were selected. The number of population members was defined by 20 times the total number of variables in the problem following the indication of the authors. This number achieves best performance with smallest number of function evaluations. A constrained non-linear multivariable function from Matlab library was utilized for the Local Search quadratic optimization.

For the empirical analysis of SVM formulations listed in previous Chapter in classification problems, several datasets were retrieved from the UCI Machine Learning Repository [Dua and Graff, 2017]:

1. **Iris flower dataset** or **Fisher's Iris dataset**: Dataset for the classification of 3 species of Iris flower (*Iris Setosa*, *Iris Versicolour* and *Iris Virginia*). Composed of 4 feature representing length and width of sepal and petal of a flower and with an equal number of instances, 50. Since the SVM formulations were implemented for binary classification, the set was divide into two set where the *iris versicolour* is present in both.

   (a) ***Iris Setosa* and *Iris Versicolour***.

   (b) ***Iris Versicolour* and *Iris Virginia***.

2. **Haberman's Survival**: Dataset of case study of survival of patients who underwent breast cancer surgery. Consists in three feature including age, year of operation, and number of positive axillary nodes detected.

3. **Wisconsin Breast Cancer Database (January 8, 1991)** [Bennett and Mangasarian, 1992]: Dataset of breast cancer clinical cases to classify cancer status as benign or malignant. Composed of 10 biological features characterized with number from 1 to 10. All instances with missing values were removed.

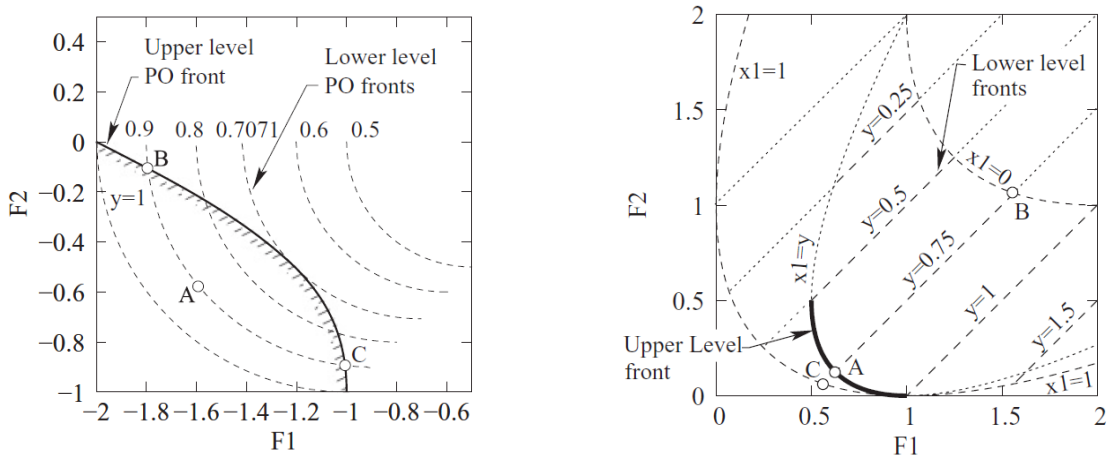
For the generalization of the classification task, the CV technique was implemented in the algorithm. For every generation, the dataset is randomized and 70% used in the lower-level training stage. The remainder is applied to validate the training results.

## 4.2 Implementation Details and Observations

In this Section, qualitative observations and remarks about the H-BLEMO algorithm and their relationship with ML problems are presented.

### 4.2.1 Behavior of Classical Multi-Objective Bi-Level Problems

Before testing ML problems in the H-BLEMO, the algorithm was simulated in classical MOBPs. This was done to evaluate the adapted algorithm and compare the results to the original algorithm. The 2 problems selected were originally from [Eichfelder, 2007] and [Deb and Sinha, 2009], and were called, respectively, TP1 and TP2 in [Deb and Sinha, 2010].

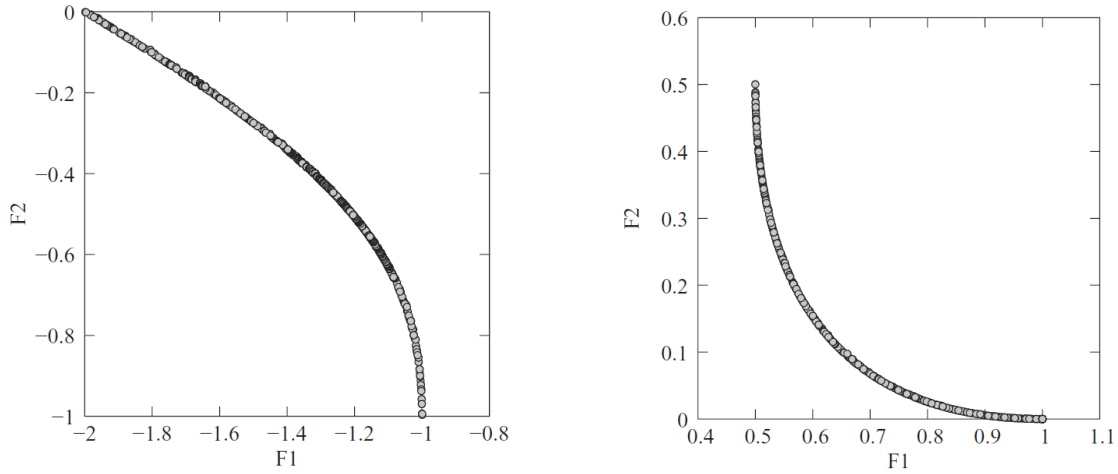



**(a)** TP1 problem

**(b)** TP2 problem

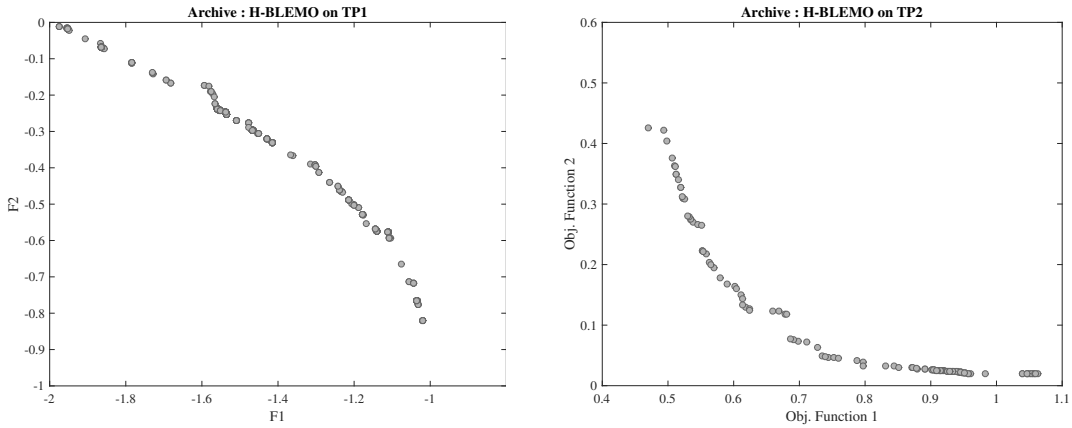
**Figure 4.1:** Pareto-optimal fronts of upper-level and some representative lower level optimization tasks are shown [Deb and Sinha, 2010].

In the objective space for both problems in figure 4.1, the upper-level PF are presented. The display of the lower-level PF helps the visualisation of the effect of constraints in the optimization solution.




**(a)** TP1 problem **(b)** TP2 problem

**Figure 4.2:** Solution of final archive of H-BLEMO [Deb and Sinha, 2010].




**(a)** TP1 problem **(b)** TP2 problem

**Figure 4.3:** Example of solution of final archive of constructed H-BLEMO.

Comparing both results of figures 4.2 and 4.3, the constructed algorithm approximately achieves the true PF but the solutions are considerably fragmented and incomplete. This indicates a lack of performance since several variable combinations are not present in the final results. In figure 4.3(b), although not evident but more clearly visible in other slightly worse runs, the algorithm has difficulty in attaining even an approximate PF. The algorithm is unable to recover the optimal PF from the PF that dominates the optimal one.

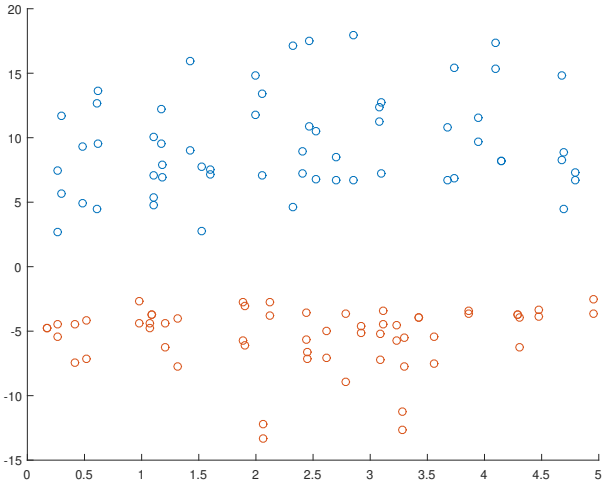
These examples had a modification on the mutation probability parameter of the algorithm. The substitution was intended to aim for more diverse results because of the poor results. However, the

parameters used in the Sections below are the same as in Section 4.1.

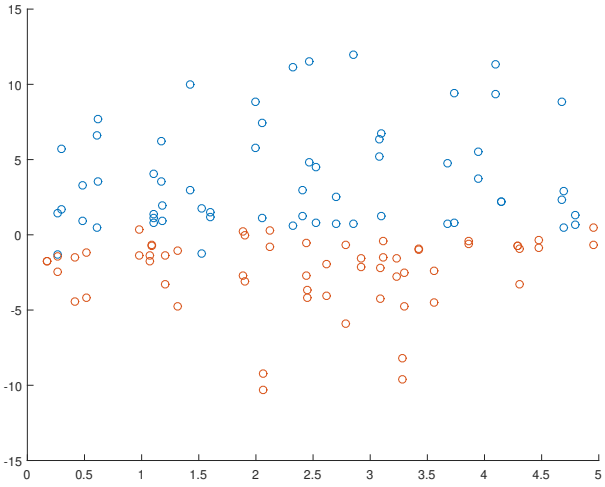
It is important to state that the H-BLEMO was constructed by the interpretation of the step procedure and the utilization of different algorithms from the original.

### 4.2.2 Upper-Level and Lower-Level Variable Vector Interval

The first predicament when starting the optimization problem lies in the definition of the range of lower-level variables. In the test problem TP1, the choice is straightforward since they are a constraint. The-




**(a)** Linearly separable

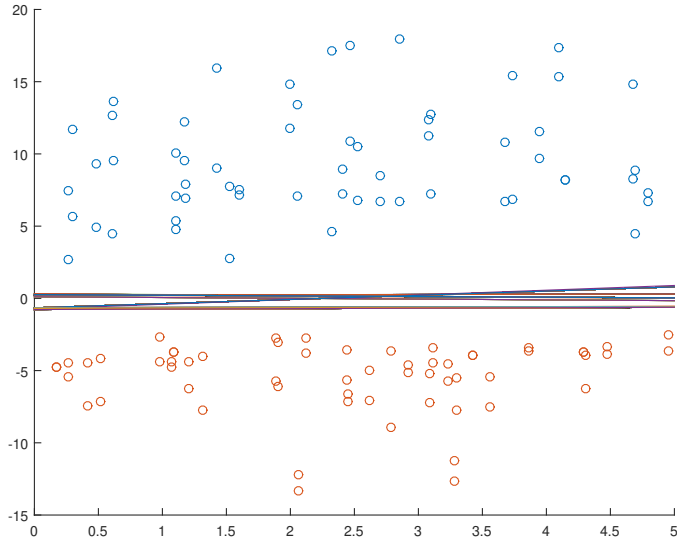



**(b)** Non-linearly separable

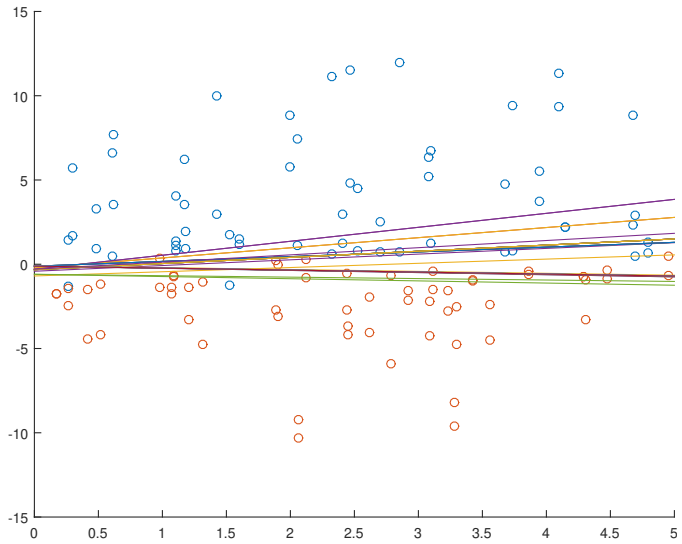
**Figure 4.4:** Basic datasets where each colour represents a different class.

oretically speaking, the parameters that define the hyperplane belong to $\mathrm{R}^n$ but, for practical reasons, the interval should be as compact as possible to achieve, simultaneously, the best results in a shorter

computation time. In respect to the upper-level, the variables are hyperparameters in the SVM problem. Therefore they should encompass large but reasonable intervals. For every test, the interval of the hyperparameter C is $[10^{-4}, 100]$. For the parameter $\sigma$ in the Gaussian transformation, the interval is defined in Section 4.2.6.




**(a)** Linearly separable dataset

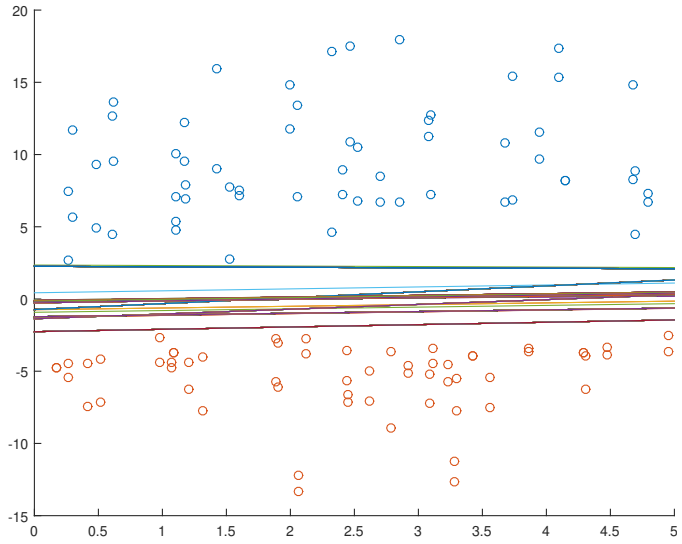



**(b)** Non-linearly separable dataset

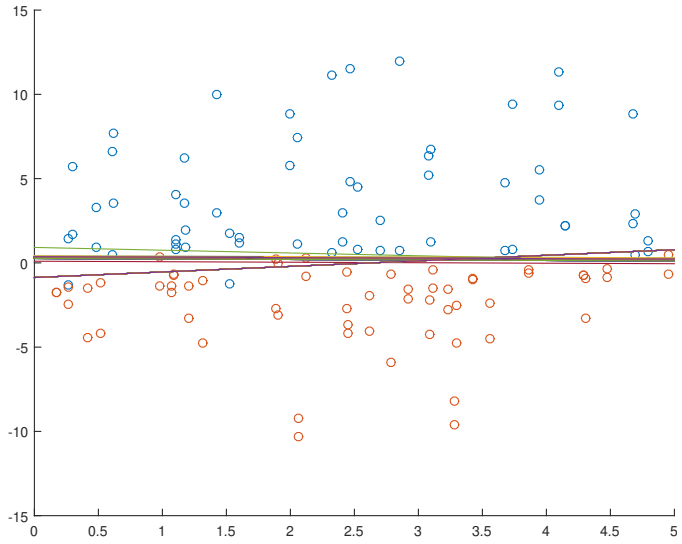
**Figure 4.5:** Effect of lower-level variable interval [-1;1] in the final archive solutions.

In the enquire of the effect of the interval in the performance of the classification, numerous tests were computed. Starting with the same interval of the TP1 problem, [-1,1], intervals were modified by a multiplying factor of 10 twice with no other change to the algorithm. Using formulation 1 the test were

performed on two simple self-made datasets, one linearly separable and the other non-linearly separable dataset, respectively presented in figure 4.4. The three tests for each dataset are presented in figures 4.5, 4.6, and 4.7.



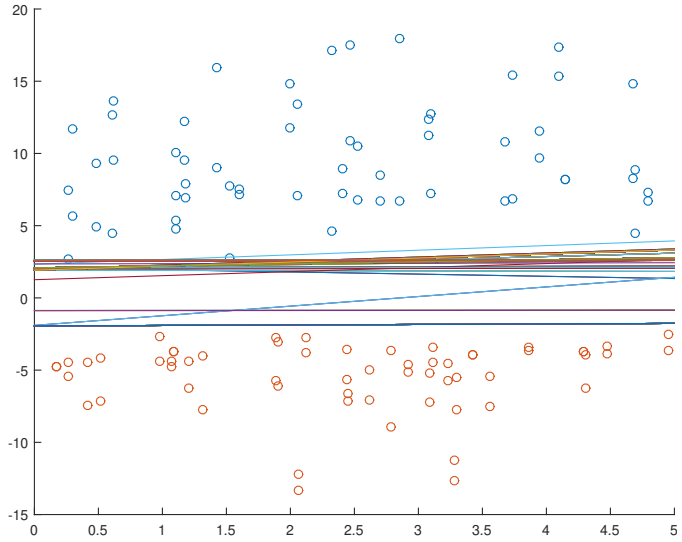

**(a)** Linearly separable dataset



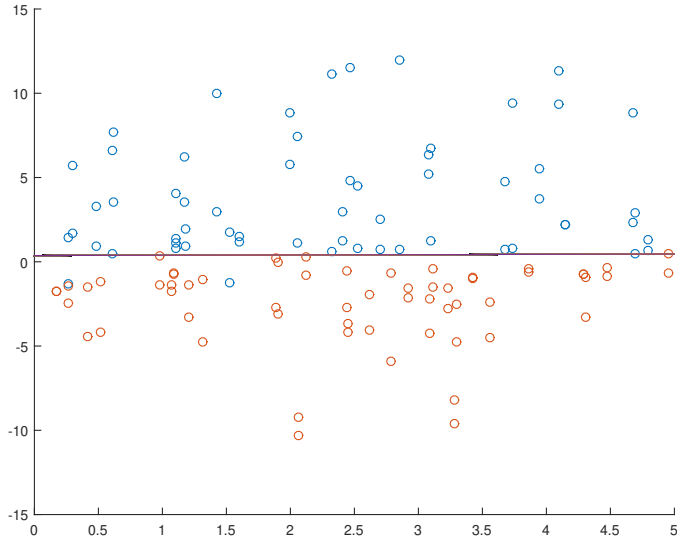

**(b)** Non-linearly separable dataset

**Figure 4.6:** Effect of lower-level variable interval [-10;10] in the final archive solutions.

In the linearly separable case, from the classification point of view, no effect is visible from the modification of the variable interval, since any hyperplane of the final archive in each test achieves perfect class separation. However, the hyperplanes become less homogeneous for the larger intervals. This outcome most probably exists due to the existence, in the feature space, of a well visible band that




**(a)** Linearly separable dataset




**(b)** Non-linearly separable dataset

**Figure 4.7:** Effect of lower-level variable interval [-100;100] in the final archive solutions.

separates both classes and the possibility of the algorithm to create a more diverse hyperplane.

Although the last observation is valuable in the generalization of the interval choice, the actual results are not observable. The test of a more demanding dataset is the primary reason for observing this effect. Since the band of the linearly separable dataset does not exist in the non-linearly example, a more concise hyperplane is demanded by the dataset for classification. Therefore the algorithm requires more freedom in selecting variables. Comparing the non-linearly separable dataset in figure 4.5, 4.6,

and 4.7, it is clear the effectiveness of a larger interval.

Of course, this decision is depending of the location of the features in the feature space, but with a simple normalization this problem can be overcome. In conclusion, the interval selected for classification test below was [-100, 100].

### 4.2.3 Upper-Level Objective Space

The upper-level is the primary optimization of the MOBP in classical problems. On the contrary, the lower-level is the main optimization in ML problems because it is the training stage of the classification task.

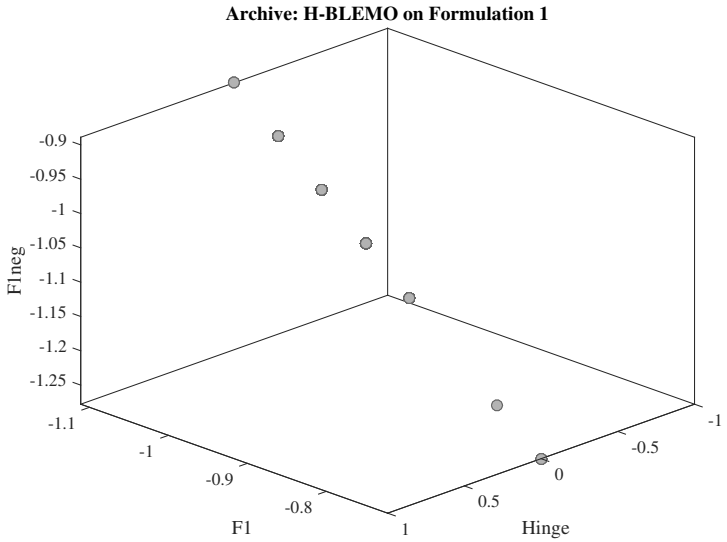
In the upper-level objective space of classical problems, the PF represents a direct outcome of the variation of the upper-level variables in the optimization. For instance, in the problem TP1 in figure 4.1(a), a particular variable $y$, $x_1$, and $x_2$ which originates a $F_1$ and $F_2$ objective value that corresponds to point C. This direct causation is the main distinction when comparing with classifications problems due to the existence of two steps in the problem: computation of a hyperplane; and classification task evaluation with the metric function(s). This means that in MOBP the hyperplane is defined by the variables and the objectives functions are defined not by the variables but by the hyperplane.

Although seemingly irrelevant, this slight difference changes the dynamic of the algorithm, for example, in the termination criteria, and the evaluation of the performance of the final results. First, the values of objective functions evaluating the classification belong to a set of finite numbers and are dependent on the number of points in the dataset and the percentage of true positive cases for the F1 score metric and true negative cases for the F1 score metric. The consequence of this is a nonsmooth and depleted PF. However, these vacant spaces in the PF are not a sign of the ineffectiveness of the algorithm but simply an intrinsic characteristic of the classification problem. Second, the points in the objective space can have the same values even though they represent different hyperplanes. The opposite can also happen, i.e., different values for the same hyperplane. These two cases take place due to not just the minimization of both the F1-scores but also the addition of the CV technique.

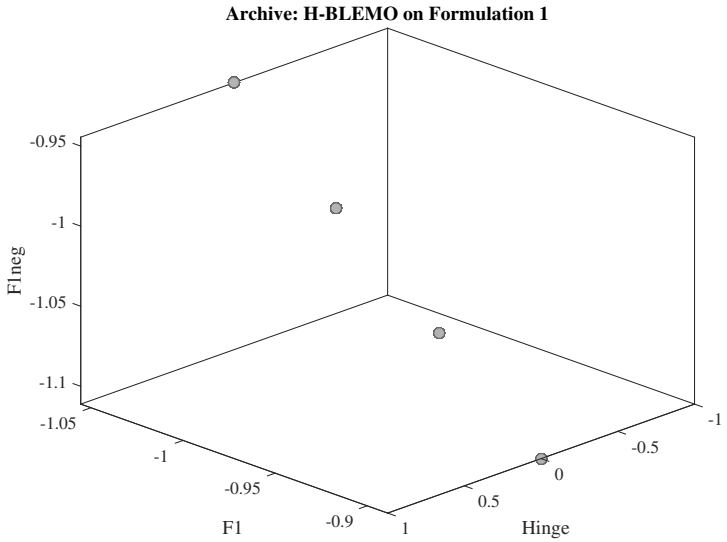
The PF can also cease to be a frontier. Using the hinge function and only the true F1 or the negative F1 instead of both, the PF becomes a single point. This occurs for two reasons: the hinge function being lower bound by 0, and the ND. Given Definition 2.2.2, when one point with hinge equal to 0 and an arbitrary F1 score value, the archive will only accept points with smaller F1 while removing the remaining.

These previous observations alter the synergy of the upper-level PF and MOBP algorithms. The objective space becomes unusable when concluding or comparing the performance of the algorithm and the different formulations. However, this does not imply an useless utility to the validation and training stage of the hyperplanes.

Using the same tests of figure 4.12 in Section 4.2.5 without taking into account the goal for using them there, the upper-level objective space was compared and are presented in figure 4.8.




**(a)** With Local Search




**(b)** Without Local Search

**Figure 4.8:** Comparison of pareto front of archive solutions of formulation 1 for the first test of Section 4.2.5.

When comparing the point in the objective space of the test in figure 4.8, the figure 4.8(a) apart from containing more points than in figure 4.8(b), only the effect of the CV technique can be visualised given the different values of F1 and negative F1 and comparing it the respective hyperplanes solutions in figure 4.12(a). The results contain hyperplanes that misclassify points whilst getting the lowest possible value of the hinge objective. This is possible to happen since the solutions represent the PF of the validation

stage of the problem, meaning only 30% of the dataset is used. In conclusion, no comparison about the performance of classification of these two test can be made.

The previous observations and conclusions made the evaluation of the advantages and disadvantages of the different formulations in the classification task from the upper-level space impossible. Also, the PF ceases to be the final solution of the problem to be the space of features. These observations were not reported in HO of Section 2.4 because the optimization was done in BP and the concept of PF is not applicable. For these reason, the original datasets were partitioned, and the new section was used for testing by computing the accuracy of solutions.

### 4.2.4 Termination Criteria

As mentioned in the previous Section, ML problems affect the MOBP, mainly the objective function space. This effect is especially concerning because of its dependency on the termination of both the lower-level and the algorithm. A comparison of the algorithm with the criteria and without was done to evaluate its impact. In replacement, a number defines the maximum number of upper-level generations.

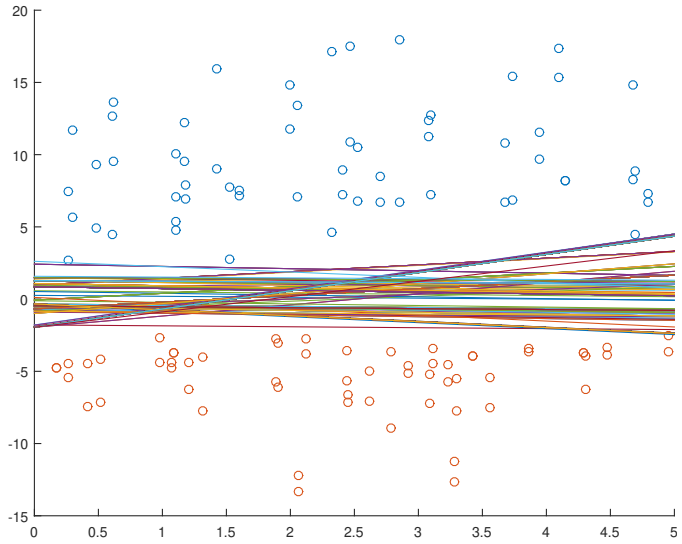
In the tests of figure 4.9, the differences are very slim, chiefly due to being an easily separable dataset. Nonetheless, the hyperplanes in figure 4.9(a) with a non-zero slope, although it separates the data, has a worse separation for future data when compared to the others.

In figure 4.10, the differences are more apparent. Some tests with the termination criteria accomplish results similar to figure 4.10(b), but not using it seems favourable to better performance. Its impact is even more expressive in more complex datasets with the presence of noise. To that effect, a new dataset was created, and the results are presented in figure 4.11.

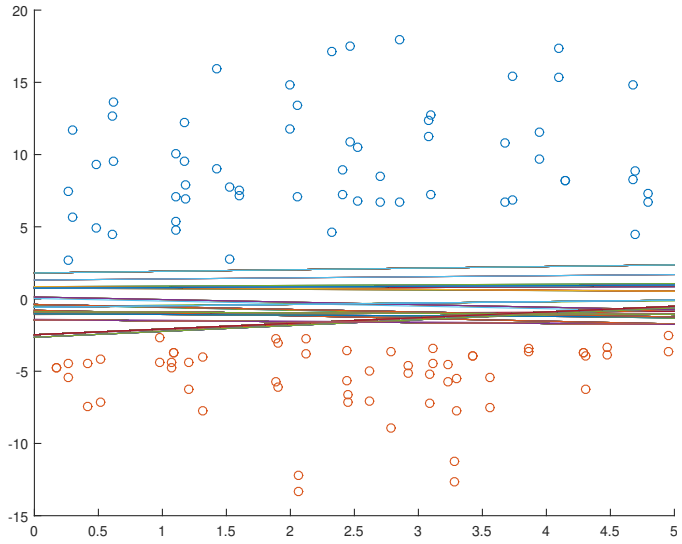
The reason for these discrepancies in the results lies in the fact that the termination criteria depends on the maximum and minimum values of the Hypervolume Indicator. In most runs of the algorithm, the upper-level PF did not change in ten consecutive generations, making both values equal and criteria zero, immediately stopping the algorithm.

Despite the identical utilization in the lower-level problem, the lower-level objective space is less prone to consecutively remaining the same due to the presence of a higher number of objective functions and in particular non-classification objectives such as, for example, the norm and the sum of the distances to hyperplane of misclassified points. For this reason and the confirmation of good performance provided by the tests without the termination criteria visible in figures 4.10(b) and 4.11(b), the lower-level did not have any changes.

In conclusion the termination criteria using the Hypervolume Indicator is not suitable for ML problems. In the tests of Section 4.3 below, it was substituted with a maximum number of generations. Two separate tests were done for every formulation and dataset with two values, 100 and 200 generations, to ensure solid results.

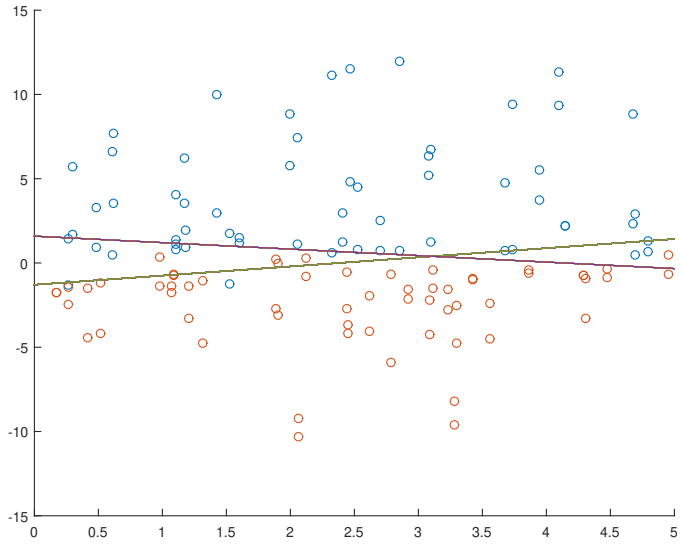



**(a)** With termination criteria

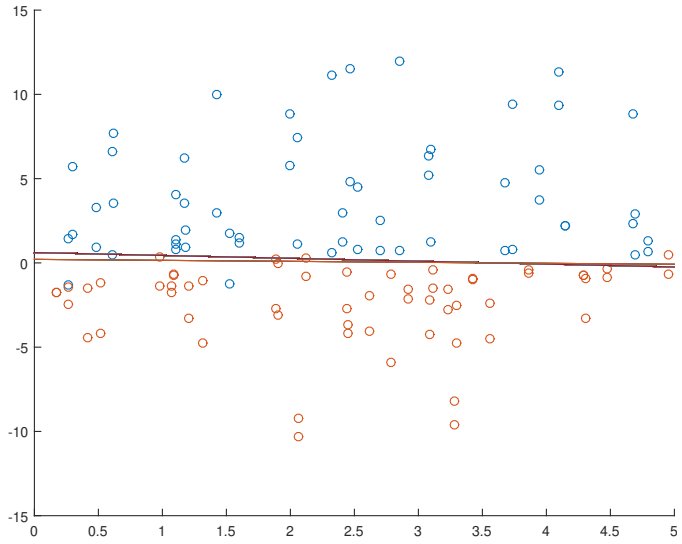



**(b)** Without termination criteria

**Figure 4.9:** Comparison of the effect of termination criteria in the final archive solutions of the linearly separable dataset with formulation 1.

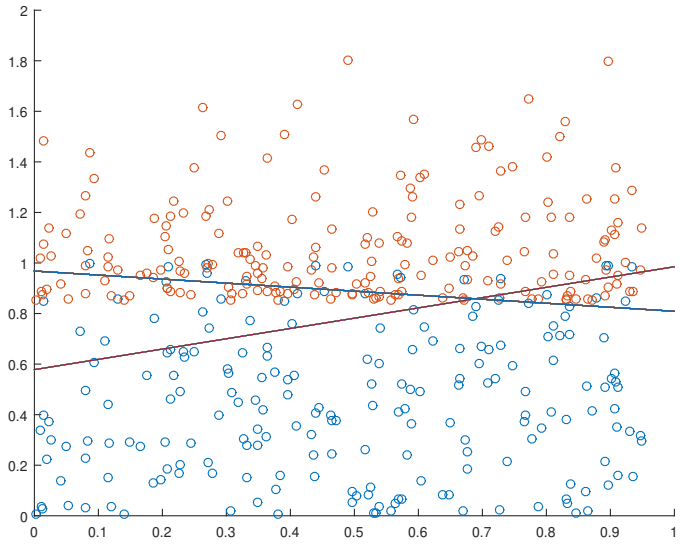



**(a)** With termination criteria

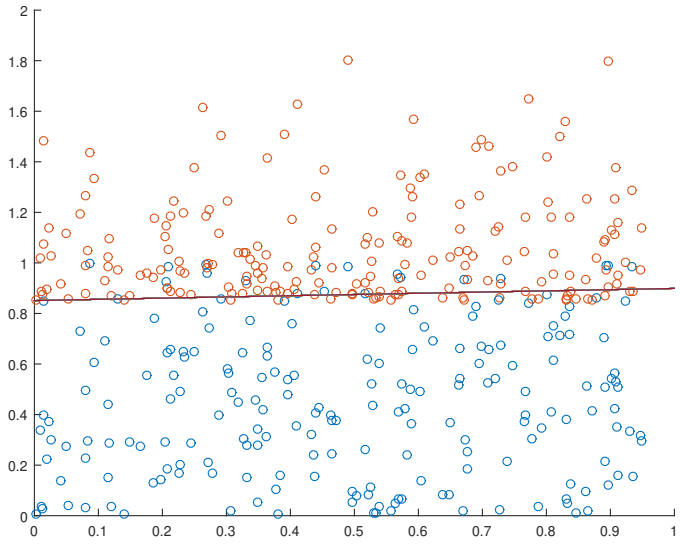



**(b)** Without termination criteria

**Figure 4.10:** Comparison of the effect of termination criteria in the final archive solutions of the non-linearly separable dataset with formulation 1.




**(a)** With termination criteria

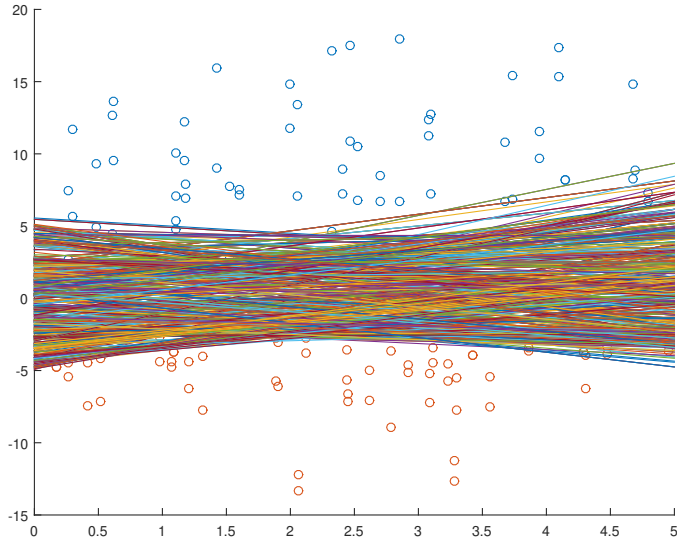



**(b)** Without termination criteria

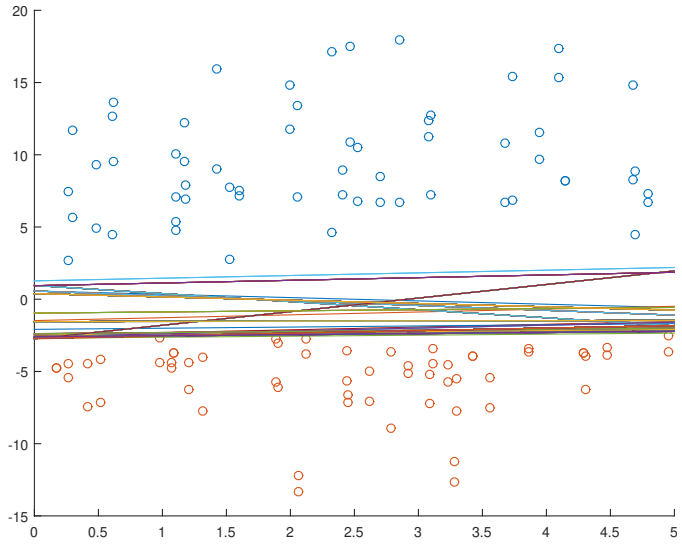
**Figure 4.11:** Comparison of the effect of termination criteria in the final archive solutions for the Noisy dataset with formulation 1.

### 4.2.5 Local Search

The Local Search optimization is the attempt to guarantee the lower-level solutions to be locally PF. ML problems change the relationship between variables, the objective function and objective space, and the problem itself. A study of the effect of the Local Search optimization was carried out. As in the previous Sections, both the linearly and non-linearly separable datasets were used. The results are presented, respectively, in figure 4.12 and figure 4.10.



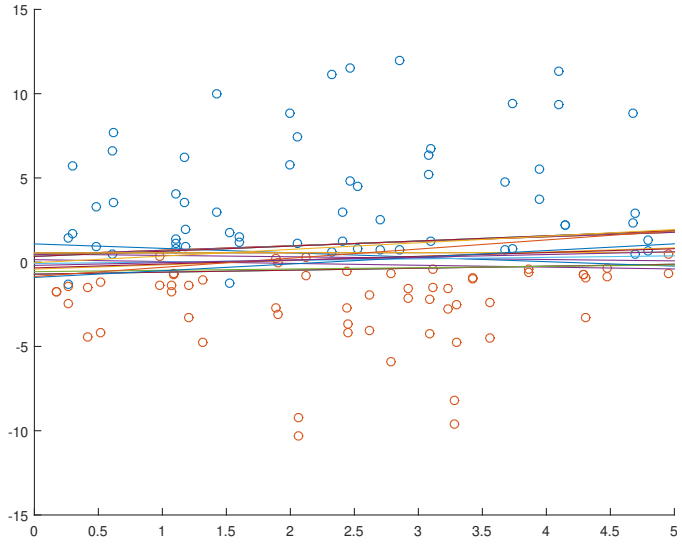

**(a)** With Local Search



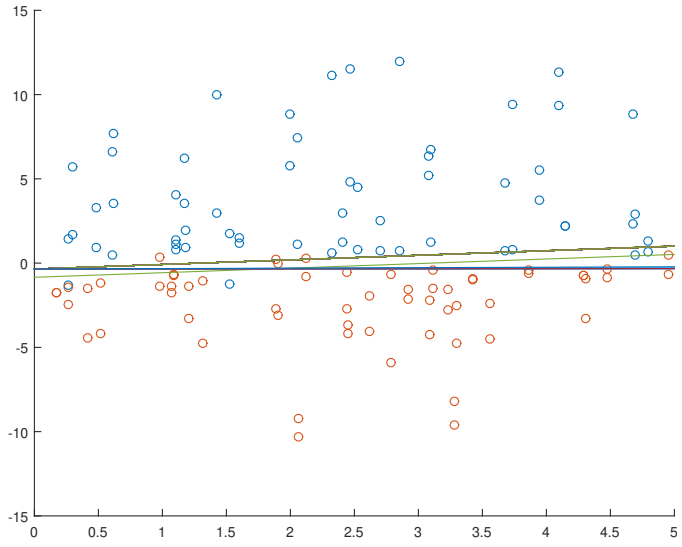

**(b)** Without Local Search

**Figure 4.12:** Comparison of the effect of Local Search in the final archive solutions for the linearly separable dataset with formulation 1.

The first observable aspect in figure 4.12(a) is how the solutions are more sparse and seemingly more diverse in terms of the number of different hyperplanes when compared with figure 4.12(b). Comparing it to figure 4.4(a), several hyperplanes do not correctly divide this trivial space. Of course, the archive contains solutions that perfectly classify the data that could be highlighted after measuring the accuracy in the test stage. However, in terms of the overall performance of the H-BLEMO, the non-perfect solutions can be detrimental in the creation step of new child solutions due to their dependency on the archive.




**(a)** With Local Search




**(b)** Without Local Search

**Figure 4.13:** Comparison of the effect of Local Search in the final archive solutions for the non-linearly separable dataset with formulation 1.

For a dataset with a more restricted space for the best separation (that is the case of the dataset in figure 4.10), the algorithm, in the long run, tends to have all solutions in the archive alike or coinciding with each other. Comparing figures 4.13(a) and 4.13(b) again it is observed that the Local Search did not aid the results.

In conclusion, the removal of the operator does not negatively affect the H-BLEMO. On the contrary, it improved or corrected the performance of solutions. Another advantage of this disposal is the significant reduction of the computation time on an average of approximately four times. The reason for these problems has two possibly origins: the algorithm constructed has worse performance when compared to the original one, and/or the choice of the Local Search algorithm and respective parameters. For these reasons and improvements, the particular Local Search algorithm was removed, but the rest of the preparation of the Local Search step was maintained.

### 4.2.6 Feature Transformation

The classifications tests in Section 4.3 used only the Gaussian transformation to accomplish better results when compared with the original dataset. The hyperparameter that characterizes this transformation impacts only the variables of the feature space, something which is not directly connected with the H-BLEMO but with the SVM formulation problem.

However, if a polynomial transformation (see table 2.1) were to be used, the hyperparameter $d$ responsible for selecting the degree will cause problems in the algorithm. The sizes range of the lower-level and upper-level variables are chosen by the user. They are purposely static given that, in the child population creation stage, the crossover and mutation operations require the exact size of vectors. Also, when conjoined with other hyperparameters, a mixture of variable encoding exist, and, for each one, the genetic operators are computed in different methods.

The interval of the hyperparameter employed is bound by $10^{-4}$ and double the maximum absolute value of all features. The upper limit is defined in this way to prevent the transformation from becoming a flat hyperplane. Also, the transformation was defined as a degree above the original dataset and centered around the average of the features.

## 4.3 Classification Results

The objective of this work is the proof of concept of using MOBP algorithms for optimization of classification problems, there was no necessity for fine tuning for each dataset. For this reason, only simpler separations were used, that is flat hyperplanes.

To evaluate the performance of the classification of the final solutions. the testing stage was created. Using 20% of the original dataset, which was initially saved, the accuracy was computed after the algo-

rithm finished to run. In the replacement of the termination criteria with the number of generations, the assurance that the algorithm would stop when it achieved good results disappeared. For this reason, two different maximum values of generations were tested. The succinct results for the dataset *Iris Setora* and *Iris Versicolour*, Noisy, *Iris Versicolour* and *Iris Virginia*, Haberman's Survival, and Wisconsin Breast Cancer Database are presented, respectively in the tables 4.1, 4.3, 4.2, 4.4, and 4.5.The extensive results will only be presented in Appendix A.

At the end of each run, the algorithm contains multiple solutions. The accuracy in percentage for each run of the tables in the Appendix represent the mean across all. In Gaussian tests, in particular, the comparison is not just done between hyperplanes but also with different $\sigma$ values. In Section 4.3.1, the values presented are only the mean of the total number of runs.

### 4.3.1 H-BLEMO Results

In table 4.1, the algorithm, independently of the formulation, has a perfect separation rate. This result is expected, as the dataset has ample space between the two classes similar to dataset in figure 4.4. Since the following datasets are more complex, the influences of the separation of different objectives in the formulations can be detected.

**Table 4.1:** Accuracy (%) and standard deviation of runs for each formulation for the *Iris Setosa* and *Iris Versicolour* dataset.

| | **Generations** | |
|---|---|---|
| | 100 | |
| **Transformation** | **Original** | |
| **Formulation** | **Mean** | **SD** |
| 1 | 99.97 | 0.0543 |
| 2 | 100 | 0 |
| 3 | 99.76 | 0.4656 |
| 4 | 100 | 0.1845 |
| 5 | 100 | 0 |
| 6 | 100 | 0.0106 |

In table 4.2, contrary to the previous one, there is no perfect flat hyperplane that separates the space. However, a slight improvement can be observed along with the different formulations in the original dataset. For this dataset, the slack objective improves the optimization when separated from the norm objective. Although in the formulation 3.4 (3) that is not verified, the presence of the surplus objective is what improves the performance in 1.5% or 2.56% in the 200 gen test, when compared with formulation 3.2 (1). The introduction of the surplus objective likewise helps the performance, notably, of formulation 3.5 (4), for which the best result is achieved of roughly 95%. As for the Gaussian transformation, the values are seemingly beneficial in formulation 3.2 (1) and 3.4 (3) but have a high enough value of

**Table 4.2:** Accuracy (%) and standard deviation of runs with different generations for each formulation for the *Iris Versicolour* and *Iris Virginia* dataset.

| | **Generations** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 100 | | | | 200 | | | |
| **Transformation** | **Original** | | **Gaussian** | | **Original** | | **Gaussian** | |
| **Formulation** | **Mean** | **SD** | **Mean** | **SD** | **Mean** | **SD** | **Mean** | **SD** |
| 1 | 91.67 | 6.1301 | 94.99 | 5.4773 | 91.43 | 5.0766 | 91.43 | 5.6057 |
| 2 | 92.00 | 4.0000 | 92.93 | 2.5386 | 92.00 | 2.4495 | 90.58 | 7.1666 |
| 3 | 93.16 | 3.8931 | 96.82 | 6.0383 | 93.99 | 3.6782 | 95.81 | 3.5094 |
| 4 | 94.93 | 1.5142 | 93.89 | 1.4285 | 95.44 | 4.1858 | 95.48 | 2.7633 |
| 5 | 94.00 | 4.8990 | 92.19 | 3.1306 | 95.02 | 3.1625 | 93.20 | 4.0513 |
| 6 | 94.01 | 3.4765 | 96.64 | 1.8939 | 94.68 | 3.0684 | 94.67 | 2.9887 |

standard deviation to indicate overfitting. However, both formulations 3.5 (4) and 3.7 (6) attain similar and slightly better results than without transformation, respectively.

**Table 4.3:** Accuracy (%) and standard deviation of runs with different generations for each formulation for the Noisy dataset (see Section 4.2.4).

| | **Generations** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 100 | | | | 200 | | | |
| **Transformation** | **Original** | | **Gaussian** | | **Original** | | **Gaussian** | |
| **Formulation** | **Mean** | **SD** | **Mean** | **SD** | **Mean** | **SD** | **Mean** | **SD** |
| 1 | 87.79 | 3.3481 | 86.84 | 5.9969 | 85.22 | 2.1326 | 88.17 | 6.1430 |
| 2 | 88.22 | 4.3825 | 89.91 | 4.5375 | 85.35 | 4.9404 | 89.35 | 3.7455 |
| 3 | 92.47 | 2.7534 | 93.02 | 2.7588 | 94.57 | 1.4614 | 92.63 | 3.6807 |
| 4 | 92.40 | 2.7183 | 95.13 | 2.3548 | 92.76 | 2.7014 | 92.36 | 0.9536 |
| 5 | 93.00 | 1.6956 | 92.24 | 2.9985 | 93.50 | 2.8940 | 92.88 | 3.7838 |
| 6 | 93.31 | 2.3762 | 89.63 | 3.0316 | 92.50 | 1.4309 | 92.80 | 3.2020 |

In table 4.3, the effects of the slack and surplus objectives are similar to the *Iris Versicolour* and *Iris Virginia* dataset, albeit the best performances in the original dataset are formulation 3.4 (3) and 3.6 (5), around 93.50% and 93.25%, respectively. With the Gaussian transformation, the improvement is barely significant. In the case of the formulation 3.5 (4), the best mean value is achieved in the 100 generation test whereas, in the 200 generation test, the performance is worse than formulation 3.4 (3), 3.5 (4) and 3.7 (6) meaning this formulation is not as good as the formulation 3.4 (3) with no transformation.

The Haberman's Survival dataset was by far the dataset with worst performance as well as the most standard deviation. The reason for this disparity is the extreme complexity of the dataset itself seen in figure 4.14 and for this reason the very difficult task of separate with flat or non-flat hyperplanes.

Despite this, the advantage of some formations is visible. Again the separation of slack objective and the introduction of the slack objective increased the overall performance. Especially, formulation 3.5 (4) in the original transformation achieves around 10% and 5% better results than formulation 3.2 (1) and 3.3 (2), respectively. The best results are the 72% of the formulation indicated above and 73% of

**Table 4.4:** Accuracy (%) and standard deviation of runs with different generations for each formulation for the Haberman's Survival dataset.

| | **Generations** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 100 | | | | 200 | | | |
| **Transformation** | **Original** | | **Gaussian** | | **Original** | | **Gaussian** | |
| **Formulation** | **Mean** | **SD** | **Mean** | **SD** | **Mean** | **SD** | **Mean** | **SD** |
| 1 | 63.37 | 3.6062 | 69.51 | 6.6618 | 65.37 | 8.2017 | 63.15 | 5.9226 |
| 2 | 67.59 | 8.3339 | 62.26 | 4.7574 | 65.90 | 6.8304 | 62.19 | 5.6920 |
| 3 | 69.70 | 5.4575 | 67.37 | 4.2309 | 69.18 | 1.7316 | 71.81 | 6.4129 |
| 4 | 72.10 | 5.5328 | 69.79 | 5.6901 | 72.89 | 4.2815 | 69.93 | 3.2024 |
| 5 | 68.83 | 8.1508 | 71.55 | 1.9678 | 71.17 | 7.4315 | 71.59 | 0.7218 |
| 6 | 73.41 | 3.4715 | 72.00 | 4.1078 | 71.82 | 4.2295 | 72.83 | 5.1015 |

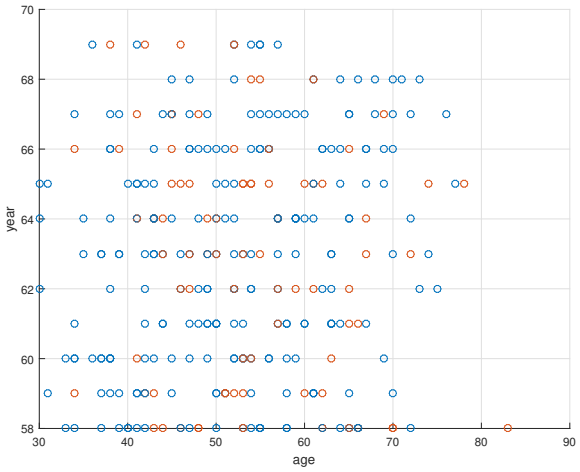
formulation 3.7 (6). In the Gaussian transformation tests, the performance is similar or worse than that of the original transformation. Comparing the tests of different generations, no advantage is found in using more than 100 generations, since no considerable reduction in the standard deviation is attained. Again, this is a consequence of the complexity of the dataset.

**Table 4.5:** Accuracy (%) and standard deviation of runs with different generations for each formulation for the Wisconsin Breast Cancer dataset.

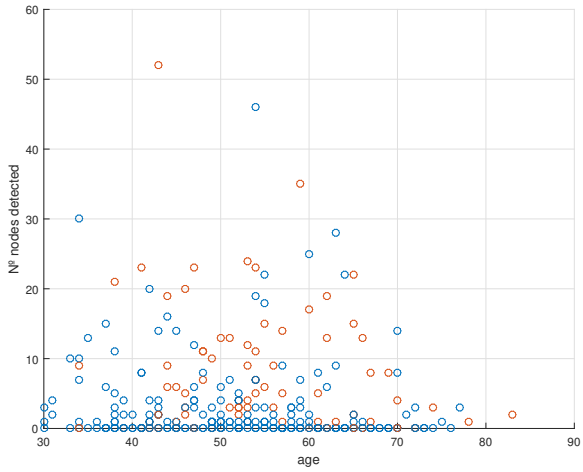
| | **Generations** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 100 | | | | 200 | | | |
| **Transformation** | **Original** | | **Gaussian** | | **Original** | | **Gaussian** | |
| **Formulation** | **Mean** | **SD** | **Mean** | **SD** | **Mean** | **SD** | **Mean** | **SD** |
| 1 | 76.46 | 9.3958 | 91.66 | 2.0571 | 78.02 | 4.6195 | 92.31 | 1.5933 |
| 2 | 67.44 | 5.5326 | 86.61 | 6.7933 | 72.26 | 6.0014 | 90.57 | 5.6901 |
| 3 | 91.96 | 0.8946 | 96.58 | 1.4128 | 93.30 | 1.8187 | 95.12 | 0.1920 |
| 4 | 97.03 | 0.5305 | 96.51 | 1.5377 | 95.72 | 1.5826 | 96.53 | 1.5827 |
| 5 | 92.71 | 3.3031 | 95.80 | 1.3211 | 95.34 | 1.0050 | 94.95 | 1.4038 |
| 6 | 95.49 | 1.5539 | 96.16 | 1.6965 | 95.87 | 1.5312 | 96.62 | 1.3957 |

In table 4.5, the results for the most challenging dataset in terms of the total number of features is presented. The dataset is the most notable for the positive influence of the introduction of the slack objective, achieving around 15% better accuracy and significantly better precision between formulation 3.2 (1) and 3.4 (3). The better results are attained when the two variable objectives are solely in the same function, in particular the best in formulation 3.5 (4) reached approximately 97% in accuracy. With the Gaussian transformation, a drastic improvement is reached in formulation 3.2 (1) and 3.3 (2) though insufficient to surpass the remaining. Also, there is no distinction in the performance when the surplus objective is introduced, yet they still achieved similar results to the best formulation in the original dataset.

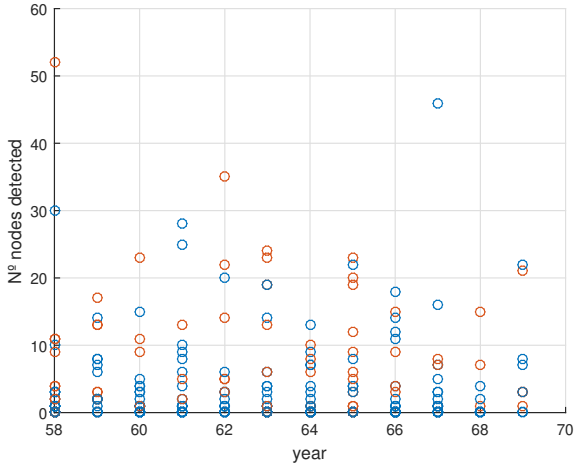
Comparing the overall datasets, the test of the *Iris Setosa* and *Iris Versicolour* had similar computational times when compared with the Wisconsin Breast Cancer dataset. Since the former is not as complex as the latter, the explanation for this is in the total number of generations. For 100, the value is




**(a)** Age and year




**(b)** Age and number of nodes




**(c)** Year and number of nodes

**Figure 4.14:** Three axes views of the Haberman's Survival dataset

simply excessive for the optimization reaching a point of repetitive comparisons of newfound solutions and the solution in the archive. Of course, the total number of possible archive members can be reduced since there is no necessity for a complete filled PF. However, the value should be chosen according to the dataset. The values used here were the same for all test simply because no prior sensibility was known.

### 4.3.2 Comparison with Dual Formulation Algorithm Results

For comparing this new approach with classical ML algorithms, tests were done with the dual formulation with mean of performance of validation data along with CV technique. The results are presented in figure

4.6. These tests were tune according to the hyperparameters just to some extent. Therefore, a more exhaustive fine-tuning could have provided better results.

**Table 4.6:** Accuracy (%) and standard deviation of test with soft margin SVM in the dual problem formulation for each dataset.

| | **Transformation** | | | |
|---|---|---|---|---|
| | **Original** | | **Gaussian** | |
| **Dataset** | **Mean** | **SD** | **Mean** | **SD** |
| *Iris Setosa* and *Iris Versicolour* | 100 | 0 | 100 | 0 |
| Noisy | 93.34 | 2.0263 | 92 | 2.0484 |
| *Iris Versicolour* and *Iris Virginia* | 95.33 | 3.3993 | 96.40 | 3.5901 |
| Haberman's Survival | 73.07 | 3.9875 | 76.46 | 3.8640 |
| Wisconsin Breast Cancer | 96.93 | 0.9491 | 97.25 | 0.8552 |

Comparing the results of table 4.6 and the best results of previous tests, the H-BLEMO achieved similar results except for the most complex dataset, the Haberman's Survival.

This proves how versatile MOBP algorithms can be in the optimization of different types of problems, as well being a competitor with traditional SVM algorithms allowing a shift in focus from also the selection of the best set of the hyperparameters to just the choice of feature transformation, the type of hyperplane and different formulations and objectives.

# 5

# Conclusions and Further Work

## Contents


5.1 Conclusions . . . . . 48

5.2 Further Work . . . . . 48

## 5.1 Conclusions

Traditional SVM algorithms, despite efficient for simple or small dataset, for large datasets and the presence of high number of hyperparameters, suffer from combinatorial explosion, making them unusable. A new alternative consisted in transforming the problem into a mathematical problem of two levels, where one is the minimization problem of the training and evaluation and the other the minimization of the validation.

For this the MOBP algorithms based on EA and their concepts were used. In particular, the algorithm used is called H-BLEMO. Since the algorithms allow optimization of multiple objectives at the same time in each level, several formulations of the SVM problem were created, focusing on two objectives: the slack objective and the surplus objective. These creations are intended to evaluate the existence of possible conflicts between objectives and the advantages of separating these objectives into different optimization functions.

In total six formulations were created. The several tests indicated that classification problems change how the MOBP is optimized and the respective conclusions of the final solution. There is no direct connection with the solution of the training problem (hyperplane) and of the PF. For this reason, the PF becomes a just a tool to achieve the best results and not to evaluate the final archive hyperplanes solutions and thus a different termination criteria is required. The overall tests indicate the introduction of the surplus objective in the formulation is preferable since the models achieved better results with it than without. The best formulation with this objective vary with the dataset. However, the results were slightly worse when this objective was by itself. As for the soft margin based formulation, the results were similar with the best formulation also vary with the dataset.

In conclusion, the utilization of MOBP to solve ML problems are a reliable alternative and a good competitor to the classical SVM algorithms, since it allows an automatize selection of hyperparameters and the testing for advantages and disadvantages of the relation between objectives and their separation.

## 5.2 Further Work

This work proved the potential of this concept. However, other studies are required for understanding the real impact of classification problems in MOBP and for fine-tuning.

First, although the performances were still good while not removing it, an analysis of the impact of the termination criteria in the lower-level optimization should be made. An analysis is also required to understand the reason for the negative effect of using the Local Search optimization.

Second, due to the changes in the objective space, an evaluation should be performed on the ranking of solutions to verify if the CD criteria are relevant and necessary, since there is no utility for scarcity in

the PF.

Finally, more tests should be done with a higher number of features ($> 10$), testing other formulations such as the remaining presented in Section 2.1, and taking into account different hyperparameters for each class in imbalanced datasets. Also, the test should include non-flat hyperplanes for separation and use the algorithm Reference-Point based Many-Objective NSGA-II in the lower-level or both levels.

# Bibliography

[Aiyoshi and Shimizu, 1981] Aiyoshi, E. and Shimizu, K. (1981). Hierarchical decentralized systems and its new solution by a barrier method. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-11(6):444–449.

[Al-Khayyal et al., 1992] Al-Khayyal, F. A., Horst, R., and Pardalos, P. M. (1992). Global optimization of concave functions subject to quadratic constraints: An application in nonlinear bilevel programming. *Ann. Oper. Res.*, 34(1–4):125–147.

[Alves and Antunes, 2017] Alves, M. J. and Antunes, C. H. (2017). An illustration of different concepts of solutions in semivectorial bilevel programming. *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*.

[Andersson et al., 2015] Andersson, M., Bandaru, S., Ng, A., and Syberfeldt, A. (2015). Parameter tuning of moeas using a bilevel optimization approach. In Gaspar-Cunha, A., Henggeler Antunes, C., and Coello, C. C., editors, *Evolutionary Multi-Criterion Optimization*, pages 233–247, Cham. Springer International Publishing.

[Andersson et al., 2016] Andersson, M., Bandaru, S., and Ng, A. H. (2016). Tuning of multiple parameter sets in evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, GECCO '16, page 533–540, New York, NY, USA. Association for Computing Machinery.

[Bengio, 2000] Bengio, Y. (2000). Gradient-Based Optimization of Hyperparameters. *Neural Computation*, 12(8):1889–1900.

[Bennett et al., 2006] Bennett, K. P., Jing Hu, Xiaoyun Ji, Kunapuli, G., and Jong-Shi Pang (2006). Model selection via bilevel optimization. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 1922–1929.

[Bennett and Mangasarian, 1992] Bennett, K. P. and Mangasarian, O. L. (1992). Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods & Software*, 1(1):23–34.

[Bergstra and Bengio, 2012] Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(null):281–305.

[Bergstra et al., 2013] Bergstra, J., Yamins, D., and Cox, D. (2013). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *ICML*.

[Carrasqueira et al., 2015] Carrasqueira, P., Alves, M. J., and Antunes, C. H. (2015). A bi-level multiobjective pso algorithm. In Gaspar-Cunha, A., Henggeler Antunes, C., and Coello, C. C., editors, *Evolutionary Multi-Criterion Optimization*, pages 263–276, Cham. Springer International Publishing.

[Claesen and Moor, 2015] Claesen, M. and Moor, B. D. (2015). Hyperparameter search in machine learning. *CoRR*, abs/1502.02127.

[Coello, 1999] Coello, C. A. C. (1999). Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems*, 1:269–308.

[Coello Coello and Lechuga, 2002] Coello Coello, C. and 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)*, volume 2, pages 1051–1056 vol.2.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

[da Silva Santos et al., 2021] da Silva Santos, C. E., Sampaio, R. C., dos Santos Coelho, L., Bestard, G. A., and Llanos, C. H. (2021). Multi-objective adaptive differential evolution for svm/svr hyperparameters selection. *Pattern Recognition*, 110:107649.

[Deb, 2010] Deb, K. (2010). *Recent Developments in Evolutionary Multi-Objective Optimization*, pages 339–368. Springer US, Boston, MA.

[Deb and Agrawal, 1995] Deb, K. and Agrawal, R. (1995). Simulated binary crossover for continuous search space. *Complex Syst.*, 9.

[Deb and Jain, 2014] Deb, K. and 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(4):577–601.

[Deb et al., 2002] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

[Deb and Sinha, 2009] Deb, K. and Sinha, A. (2009). Solving bilevel multi-objective optimization problems using evolutionary algorithms. In Ehrgott, M., Fonseca, C. M., Gandibleux, X., Hao, J.-K., and

Sevaux, M., editors, *Evolutionary Multi-Criterion Optimization*, pages 110–124, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Deb and Sinha, 2010] Deb, K. and Sinha, A. (2010). An efficient and accurate solution methodology for bilevel multi-objective programming problems using a hybrid evolutionary-local-search algorithm. *Evolutionary Computation*, 18(3):403–449. PMID: 20560758.

[Dempe et al., 2006] Dempe, S., Dutta, J., and Lohse, S. (2006). Optimality conditions for bilevel programming problems. *Optimization*, 55(5-6).

[Dua and Graff, 2017] Dua, D. and Graff, C. (2017). UCI machine learning repository.

[Eichfelder, 2007] Eichfelder, G. (2007). *Solving nonlinear multiobjective bilevel optimization problems with coupled upper level constraints*. Inst. für Angewandte Mathematik.

[Fischer et al., 2015] Fischer, A., Langensiepen, G., Luig, K., Strasdat, N., and Thies, T. (2015). Efficient optimization of hyper-parameters for least squares support vector regression. *Optimization Methods Software*, 30(6):1095–1108.

[Fonseca et al., 2017] Fonseca, C. M., López-Ibáñez, M., Paquete, L., and Geurreiro, A. P. (2017). Computation of the hypervolume indicator. http://lopez-ibanez.eu/hypervolume. Accessed: 18-01-2021.

[Fonseca et al., 2006] Fonseca, C. M., Paquete, L., and Lopez-Ibanez, M. (2006). An improved dimension-sweep algorithm for the hypervolume indicator. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1157–1163.

[Hong-Gunn Chew et al., 2001] Hong-Gunn Chew, Bogner, R. E., and Cheng-Chew Lim (2001). Dual /spl nu/-support vector machine with error rate and training size biasing. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 2, pages 1269–1272 vol.2.

[Igel, 2005] Igel, C. (2005). Multi-objective model selection for support vector machines. In Coello Coello, C. A., Hernández Aguirre, A., and Zitzler, E., editors, *Evolutionary Multi-Criterion Optimization*, pages 534–546, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Ishibuchi et al., 2016] Ishibuchi, H., Imada, R., Setoguchi, Y., and Nojima, Y. (2016). Performance comparison of nsga-ii and nsga-iii on various many-objective test problems. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 3045–3052.

[Ishizuka and Aiyoshi, 1992] Ishizuka, Y. and Aiyoshi, E. (1992). Double penalty method for bilevel optimization problems. *Annals of Operations Research*, 34(1):73–88.

[Jain and Deb, 2014] Jain, H. and Deb, K. (2014). An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18(4):602–622.

[Kukkonen and Deb, 2006] Kukkonen, S. and Deb, K. (2006). Improved pruning of non-dominated solutions based on crowding distance for bi-objective optimization problems. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1179–1186.

[Kunapuli et al., 2012] Kunapuli, G., Bennett, K., Hu, J., and Pang, J.-S. (2012). Bilevel model selection for support vector machines. *CRM Proceedings and Lecture Notes*, 45.

[Lalwani et al., 2013] Lalwani, S., Singhal, S., Kumar, R., and Gupta, N. (2013). A comprehensive survey: Multi-objective particle swarm optimization (mopso) algorithm: Variants and applications. *Transactions on Combinatorics*, 2:89–101.

[Min Yoon et al., 2003] Min Yoon, Yeboon Yun, and Nakayama, H. (2003). A role of total margin in support vector machines. In *Proceedings of the International Joint Conference on Neural Networks, 2003.*, volume 3, pages 2049–2053 vol.3.

[Moore et al., 2011] Moore, G., Bergeron, C., and Bennett, K. (2011). Model selection for primal svm. *Machine Learning*, 85:175–208.

[Morik et al., 1999] Morik, K., Brockhausen, P., and Joachims, T. (1999). Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, page 268–277, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Musicant et al., 2003] Musicant, D. R., Kumar, V., and Ozgur, A. (2003). Optimizing f-measure with support vector machines. In *FLAIRS Conference*, pages 356–360.

[Okuno et al., 2018] Okuno, T., Takeda, A., and Kawana, A. (2018). Hyperparameter learning via bilevel nonsmooth optimization.

[Santiago et al., 2014] Santiago, A., Huacuja, H. J. F., Dorronsoro, B., Pecero, J. E., Santillan, C. G., Barbosa, J. J. G., and Monterrubio, J. C. S. (2014). *A Survey of Decomposition Methods for Multi-objective Optimization*, pages 453–465. Springer International Publishing, Cham.

[Schaffer, 1985] Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, page 93–100, USA. L. Erlbaum Associates Inc.

[Schölkopf et al., 1998] Schölkopf, B., Bartlett, P., Smola, A., and Williamson, R. (1998). Support vector regression with automatic accuracy control. In Niklasson, L., Bodén, M., and Ziemke, T., editors, *ICANN 98*, pages 111–116, London. Springer London.

[Schölokopf et al., 2000] Schölokopf, B., Smola, A. J., Williamson, R. C., and Bartlett, P. L. (2000). New support vector algorithms. *Neural Computation*, 12(5):1207–1245.

[Schutze et al., 2011] Schutze, O., Lara, A., and Coello, C. A. C. (2011). On the influence of the number of objectives on the hardness of a multiobjective optimization problem. *IEEE Transactions on Evolutionary Computation*, 15(4):444–455.

[Shimizu and Aiyoshi, 1981] Shimizu, K. and Aiyoshi, E. (1981). A new computational method for Stackelberg and min-max problems by use of a penalty method. *IEEE Transactions on Automatic Control*, 26(2):460–466.

[Sinha et al., 2020] Sinha, A., Khandait, T., and Mohanty, R. (2020). A gradient-based bilevel optimization approach for tuning hyperparameters in machine learning.

[Sinha et al., 2018] Sinha, A., Malo, P., and Deb, K. (2018). A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295.

[Sinha et al., 2014] Sinha, A., Malo, P., Xu, P., and Deb, K. (2014). A bilevel optimization approach to automated parameter tuning. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, GECCO '14, page 847–854, New York, NY, USA. Association for Computing Machinery.

[Sloss and Gustafson, 2020] Sloss, A. N. and Gustafson, S. (2020). 2019 evolutionary algorithms review. In *Genetic Programming Theory and Practice XVII*, pages 307–344. Springer International Publishing.

[Srinivas and Deb, 1994] Srinivas, N. and Deb, K. (1994). Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248.

[Tian et al., 2017] Tian, Y., Cheng, R., Zhang, X., and Jin, Y. (2017). PlatEMO: A MATLAB platform for evolutionary multi-objective optimization. *IEEE Computational Intelligence Magazine*, 12(4):73–87.

[van Rijsbergen, 1979] van Rijsbergen, C. (1979). Information retrieval.

[Vapnik, 1995] Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag New York.

[Vicente et al., 1994] Vicente, L., Savard, G., and Júdice, J. (1994). Descent approaches for quadratic bilevel programming. *J. Optim. Theory Appl.*, 81(2):379–399.

[Wierzbicki, 1980] Wierzbicki, A. P. (1980). The use of reference objectives in multiobjective optimization. In Fandel, G. and Gal, T., editors, *Multiple Criteria Decision Making Theory and Application*, pages 468–486, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Yoon et al., 2003] Yoon, M., Nakayama, H., and Yun, Y. (2003). A soft margin algorithm controlling tolerance directly. In *Multi-Objective Programming and Goal Programming*, pages 281–287, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Zhang and Li, 2007] Zhang, Q. and Li, H. (2007). Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731.

[Zitzler et al., 2000] Zitzler, E., Deb, K., and Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195.

# Accuracy Results

**Table A.1:** Noisy Dataset: Accuracy (%) of archive solutions for each run, and average and standard deviation of all runs for 100 and 200 generations for original and gaussian transformation.

| | | | | **Generations** | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 100 | | | | | | | 200 | | | | | | |
| | | | | **Run** | | | | | **Mean** | **SD** | **Run** | | | | | **Mean** | **SD** |
| | **Formulation** | **Variables** | **N** | 1 | 2 | 3 | 4 | 5 | | | 1 | 2 | 3 | 4 | 5 | | |
| **Original** | 1 | 4 | 80 | 87.43 | 89.97 | 91.25 | 82.5 | 82.50 | 87.79 | 3.3481 | 86.13 | 86.20 | 84.99 | 81.26 | 87.50 | 85.22 | 2.1326 |
| | 2 | 3 | 60 | 81.24 | 95.00 | 88.75 | 88.75 | 87.36 | 88.22 | 4.3825 | 88.24 | 90.00 | 76.01 | 87.50 | 85.00 | 85.35 | 4.9404 |
| | 3 | 5 | 100 | 88.75 | 91.25 | 96.25 | 91.10 | 95.00 | 92.47 | 2.7534 | 92.50 | 96.21 | 94.12 | 96.25 | 93.75 | 94.57 | 1.4614 |
| | 4 | 5 | 100 | 92.50 | 93.90 | 96.55 | 88.86 | 90.18 | 92.40 | 2.7183 | 95.00 | 91.21 | 96.35 | 92.50 | 88.74 | 92.76 | 2.7014 |
| | 5 | 3 | 60 | 91.25 | 91.25 | 95.00 | 92.50 | 95.00 | 93.00 | 1.69556 | 92.50 | 95.00 | 93.75 | 88.75 | 97.50 | 93.50 | 2.8940 |
| | 6 | 4 | 80 | 88.75 | 95.38 | 93.70 | 93.72 | 95.00 | 93.31 | 2.3762 | 92.50 | 94.77 | 94.85 | 96.13 | 92.50 | 94.15 | 1.4309 |
| **Gaussian** | 1 | 6 | 120 | 81.33 | 95.18 | 84.07 | 80.80 | 92.82 | 86.84 | 5.9969 | 91.04 | 75.97 | 92.28 | 91.49 | 90.08 | 88.17 | 6.1430 |
| | 2 | 5 | 100 | 91.28 | 88.02 | 89.13 | 83.64 | 97.50 | 89.91 | 4.5375 | 84.51 | 87.50 | 93.75 | 87.24 | 93.77 | 89.35 | 3.7455 |
| | 3 | 7 | 140 | 90.50 | 97.28 | 94.96 | 92.41 | 89.95 | 93.02 | 2.7588 | 92.33 | 86.74 | 91.20 | 97.40 | 95.48 | 92.63 | 3.6807 |
| | 4 | 7 | 140 | 94.57 | 9302 | 92.50 | 98.75 | 96.84 | 95.13 | 2.3548 | 92.39 | 92.50 | 90.75 | 93.75 | 92.40 | 92.36 | 0.9536 |
| | 5 | 5 | 100 | 95.00 | 87.50 | 92.48 | 95.68 | 90.52 | 92.24 | 2.9984 | 94.11 | 92.44 | 97.88 | 86.23 | 93.75 | 92.88 | 3.7838 |
| | 6 | 6 | 120 | 84.47 | 88.98 | 93.75 | 90.96 | 90.00 | 89.63 | 3.0316 | 90.68 | 95.00 | 87.50 | 95.80 | 95.00 | 92.80 | 3.2020 |

**Table A.2:** *Iris Versicolour* and *Iris Virginia Dataset*: Accuracy (%) of archive solutions for each run, and average and standard deviation of all runs for 100 and 200 generations for original and gaussian transformation.

| | | | | **Generations** | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 100 | | | | | | | 200 | | | | | | |
| | | | | **Run** | | | | | **Mean** | **SD** | **Run** | | | | | **Mean** | **SD** |
| | **Formulation** | **Variables** | **N** | 1 | 2 | 3 | 4 | 5 | | | 1 | 2 | 3 | 4 | 5 | | |
| **Original** | 1 | 6 | 120 | 100 | 85.69 | 97.64 | 90.00 | 95.02 | 91.67 | 6.1301 | 90.00 | 100 | 88.64 | 85.00 | 93.50 | 91.43 | 5.0766 |
| | 2 | 5 | 100 | 90.00 | 90.00 | 100 | 90.00 | 90.00 | 92.00 | 4.0000 | 90.00 | 95.00 | 95.00 | 90.00 | 90.00 | 92.00 | 2.4495 |
| | 3 | 7 | 140 | 91.18 | 89.44 | 90.29 | 100 | 94.89 | 93.16 | 3.8931 | 94.99 | 94.85 | 91.57 | 88.81 | 99.75 | 93.99 | 3.6782 |
| | 4 | 7 | 140 | 95.81 | 95.01 | 95.98 | 91.99 | 95.89 | 94.93 | 1.5142 | 99.89 | 94.09 | 98.58 | 88.04 | 96.62 | 95.44 | 4.1858 |
| | 5 | 5 | 100 | 100 | 95.00 | 95.00 | 95.00 | 85.00 | 94.00 | 4.8990 | 95.00 | 100 | 90.00 | 95.09 | 95.00 | 95.02 | 3.1625 |
| | 6 | 6 | 120 | 88.43 | 93.75 | 93.37 | 95.33 | 99.21 | 94.01 | 3.4765 | 95.03 | 90.07 | 94.79 | 99.68 | 93.82 | 94.68 | 3.0684 |
| **Gaussian** | 1 | 8 | 160 | 94.96 | 85.00 | 95.00 | 100 | 100 | 94.99 | 5.4773 | 93.46 | 85.02 | 93.32 | 99.94 | 85.44 | 91.43 | 5.6057 |
| | 2 | 7 | 140 | 95.00 | 89.61 | 90.03 | 95.00 | 94.99 | 92.93 | 2.5386 | 95.00 | 99.70 | 90.00 | 78.20 | 90.00 | 90.58 | 7.1666 |
| | 3 | 9 | 180 | 100 | 84.74 | 99.52 | 99.83 | 99.99 | 96.82 | 6.0383 | 100 | 94.68 | 95.35 | 88.95 | 95.05 | 94.91 | 3.5094 |
| | 4 | 9 | 180 | 95.10 | 94.41 | 91.37 | 93.34 | 95.24 | 93.89 | 1.4285 | 97.97 | 90.74 | 94.40 | 98.33 | 95.97 | 95.48 | 2.7633 |
| | 5 | 7 | 140 | 95.98 | 90.18 | 94.90 | 92.53 | 87.38 | 92.19 | 3.1306 | 92.95 | 99.85 | 89.90 | 88.33 | 94.96 | 93.20 | 4.0513 |
| | 6 | 8 | 160 | 96.53 | 99.95 | 94.95 | 97.09 | 94.67 | 96.64 | 1.8939 | 90.81 | 95.03 | 99.82 | 94.79 | 92.90 | 94.67 | 2.9887 |

**Table A.3:** Haberman's Survival Dataset: Accuracy (%) of archive solutions for each run, and average and standard deviation of all runs for 100 and 200 generations for original and gaussian transformation.

| | | | | **Generations** | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 100 | | | | | | | 200 | | | | | | |
| | | | | **Run** | | | | | **Mean** | **SD** | **Run** | | | | | **Mean** | **SD** |
| | **Formulation** | **Variables** | **N** | 1 | 2 | 3 | 4 | 5 | | | 1 | 2 | 3 | 4 | 5 | | |
| **Original** | 1 | 5 | 100 | 65.57 | 65.63 | 56.89 | 62.08 | 66.72 | 63.37 | 3.6062 | 78.50 | 64.96 | 68.76 | 53.97 | 60.64 | 65.37 | 8.2017 |
| | 2 | 4 | 80 | 70.50 | 67.21 | 54.10 | 55.74 | 75.41 | 64.59 | 8.3339 | 67.21 | 73.77 | 72.13 | 60.66 | 55.74 | 65.90 | 6.8304 |
| | 3 | 6 | 120 | 80.08 | 66.07 | 68.36 | 69.37 | 64.60 | 69.70 | 5.4575 | 69.79 | 71.43 | 69.03 | 69.53 | 66.12 | 69.18 | 1.7316 |
| | 4 | 6 | 120 | 75.70 | 65.97 | 69.42 | 81.09 | 68.31 | 72.10 | 5.5328 | 70.94 | 71.70 | 81.31 | 69.35 | 71.15 | 72.89 | 4.2815 |
| | 5 | 4 | 80 | 55.81 | 66.98 | 80.33 | 67.26 | 73.77 | 68.83 | 8.1508 | 59.08 | 68.78 | 81.94 | 72.13 | 73.89 | 71.17 | 7.4315 |
| | 6 | 5 | 100 | 67.36 | 74.14 | 78.00 | 74.73 | 72.83 | 73.41 | 3.4715 | 71.10 | 79.31 | 72.40 | 66.50 | 69.77 | 71.82 | 4.2295 |
| **Gaussian** | 1 | 7 | 140 | 61.33 | 64.97 | 73.99 | 67.31 | 79.77 | 69.51 | 6.6618 | 71.83 | 57.09 | 55.90 | 65.43 | 65.50 | 63.15 | 5.9226 |
| | 2 | 6 | 120 | 63.65 | 59.27 | 68.82 | 54.94 | 64.64 | 92.26 | 4.6574 | 61.37 | 66.36 | 63.55 | 51.74 | 67.94 | 62.19 | 5.6920 |
| | 3 | 8 | 160 | 93.99 | 64.51 | 75.61 | 66.50 | 66.21 | 67.37 | 4.2309 | 80.16 | 72.35 | 71.57 | 60.48 | 74.50 | 71.81 | 6.4129 |
| | 4 | 8 | 160 | 78.17 | 64.28 | 62.63 | 71.26 | 72.61 | 69.79 | 5.6901 | 75.00 | 64.89 | 69.52 | 70.20 | 70.00 | 69.63 | 3.2034 |
| | 5 | 6 | 120 | 70.52 | 68.39 | 73.20 | 71.74 | 73.89 | 71.55 | 1.9648 | 72.05 | 72.13 | 70.24 | 71.42 | 72.10 | 71.59 | 0.7218 |
| | 6 | 7 | 140 | 77.66 | 66.64 | 69.18 | 75.75 | 70.78 | 72.00 | 4.1078 | 78.27 | 65.99 | 79.19 | 69.62 | 71.07 | 72.83 | 5.1015 |

**Table A.4:** Wisconsin Breast Cancer Dataset: Accuracy (%) of archive solutions for each run, and average and standard deviation of all runs for 100 and 200 generations for original and gaussian transformation.

| | | | | **Generations** | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 100 | | | | | | | 200 | | | | | | |
| | | | | **Run** | | | | | **Mean** | **SD** | **Run** | | | | | **Mean** | **SD** |
| | **Formulation** | **Variables** | **N** | 1 | 2 | 3 | 4 | 5 | | | 1 | 2 | 3 | 4 | 5 | | |
| **Original** | 1 | 11 | 220 | 86.54 | 70.07 | 76.20 | 62.64 | 86.86 | 76.46 | 9.3985 | 79.60 | 83.96 | 81.28 | 72.27 | 72.99 | 78.02 | 4.6195 |
| | 2 | 10 | 200 | 73.72 | 59.85 | 72.26 | 62.04 | 69.34 | 67.44 | 5.5326 | 72.26 | 75.18 | 81.02 | 62.77 | 70.07 | 72.26 | 6.0014 |
| | 3 | 12 | 240 | 92.61 | 93.31 | 91.10 | 91.83 | 90.96 | 91.96 | 0.8946 | 94.98 | 91.97 | 94.60 | 90.36 | 94.57 | 93.30 | 1.8187 |
| | 4 | 12 | 240 | 96.93 | 97.14 | 97.27 | 97.71 | 96.10 | 97.03 | 0.5305 | 97.96 | 94.96 | 95.79 | 96.27 | 93.69 | 95.72 | 1.5826 |
| | 5 | 10 | 200 | 95.71 | 91.97 | 95.62 | 86.72 | 93.52 | 92.71 | 3.3031 | 96.35 | 96.42 | 95.62 | 94.16 | 94.16 | 95.34 | 1.0050 |
| | 6 | 11 | 220 | 94.09 | 97.68 | 93.90 | 97.00 | 94.80 | 95.49 | 1.5539 | 94.42 | 96.24 | 94.00 | 98.25 | 96.46 | 95.87 | 1.5312 |
| **Gaussian** | 1 | 13 | 260 | 89.11 | 90.48 | 92.70 | 90.94 | 95.07 | 91.66 | 2.0571 | 90.60 | 90.36 | 92.79 | 93.29 | 94.50 | 92.31 | 1.5933 |
| | 2 | 12 | 240 | 89.67 | 80.33 | 92.99 | 76.76 | 93.27 | 86.61 | 6.7933 | 95.98 | 94.89 | 80.51 | 93.28 | 88.21 | 90.57 | 5.9601 |
| | 3 | 14 | 280 | 97.74 | 95.64 | 96.39 | 94.61 | 98.54 | 96.58 | 1.4128 | 94.87 | 95.27 | 95.18 | 95.35 | 94.91 | 95.12 | 0.1920 |
| | 4 | 14 | 280 | 97.61 | 96.98 | 95.96 | 98.17 | 93.80 | 96.51 | 1.5377 | 97.16 | 97.99 | 96.70 | 93.47 | 97.33 | 96.53 | 1.5827 |
| | 5 | 12 | 240 | 93.92 | 94.67 | 97.44 | 96.86 | 96.13 | 95.80 | 1.3211 | 96.57 | 94.40 | 93.05 | 96.58 | 94.14 | 94.95 | 1.4038 |
| | 6 | 13 | 260 | 98.19 | 94.65 | 93.70 | 97.14 | 97.13 | 96.16 | 1.6965 | 96.53 | 97.18 | 97.58 | 93.97 | 97.84 | 96.62 | 1.3957 |