La borsa di dottorato è stata cofinanziata con risorse del
Programma Operativo Nazionale Ricerca e Innovazione 2014-202 (CCI 2014IT16M2OP005)
Fondo Sociale Europeo, Azione I.1 “Dottorati Innovativi con caratterizzazione Industriale”

UNIVERSITA’ DELLA CALABRIA

Dipartimento di Ingegneria Informatica, Modellistica, Elettronica e Sistemistica - DIMES

Dottorato di Ricerca in

Information and Communication Technologies (ICT)

CICLO

XXXV

Feature Selection in Classification by means of Optimization and Multi-Objective Optimization

Settore Scientifico Disciplinare MAT/09

Coordinatore: Ch.mo Prof. Giancarlo Fortino

Firma _____

Supervisore/Tutor: Ch.mo Prof. Manlio Gaudioso

Firma Firma oscurata in base alle linee guida del Garante della privacy

Dottorando: Dott./ssa Behzad Pirouz

Firma _____

Firma oscurata in base alle linee guida del Garante della privacy

Feature Selection in Classification by means of Optimization and Multi-Objective Optimization

Tesi di Dottorato

Behzad Pirouz

Preface

Acknowledgments:
I want to thank those who supported me during my PhD. In particular, a special thanks to my scientific supervisor, Prof. Manlio Gaudioso, who has supported my project during my PhD, improved my knowledge, created a pleasant work environment, and allowed me to progress during this path.
Further, I would like to thank my wife, Negar, who encouraged and supported me from the beginning of this path.
Finally, I would like to thank all my colleagues in the DIMES department.

Rende (Italy), *Behzad Pirouz*
November 2022

Abstract:
The thesis is in the area of mathematical optimization with application to Machine Learning. The focus is on Feature Selection (FS) in the framework of binary classification via Support Vector Machine paradigm. We concentrate on the use of sparse optimization techniques, which are widely considered as the election tool for tackling FS. We study the problem both in terms of single and multi-objective optimization.
We propose first a novel Mixed-Integer Nonlinear Programming (MINLP) model for sparse optimization based on the polyhedral k-norm. We introduce a new way to take into account the k-norm for sparse optimization by setting a model based on fractional programming (FP). Then we address the continuous relaxation of the problem, which is reformulated via a DC (Difference of Convex) decomposition.
On the other hand, designing supervised learning systems, in general, is a multi-objective problem. It requires finding appropriate trade-offs between several objectives, for example, between the number of misclassified training data (minimizing the squared error) and the number of nonzero elements separating the hyperplane (minimizing the number of nonzero elements). When we deal with multi-objective optimization problems, the optimization problem has yet to have a single solution that represents the best solution for all objectives simultaneously. Consequently, there is not a single solution but a set of solutions, known as the Pareto-optimal solutions.
We overview the SVM models and the related Feature Selection in terms of multi-objective optimization. Our multi-objective approach considers two simultaneous objectives: minimizing the squared error and minimizing the number of nonzero elements of the normal vector of the separator hyperplane. In this thesis, we propose a multi-objective model for sparse optimization. Our primary purpose is to demonstrate the advantages of considering SVM models as multi-objective optimization problems. In multi-objective cases, we can obtain a set of Pareto optimal solutions instead of one in single-objective cases.
Therefore, our main contribution in this thesis is of two levels: first, we propose a new model for sparse optimization based on the polyhedral k-norm for SVM classification, and second, use multi-objective optimization to consider this new model. The results of several numerical experiments on some classification datasets are reported. We used all the datasets for single-objective and multi-objective models.

Contents

1

Introduction

Machine learning is concerned with developing computer techniques and algorithms that can learn [1]. Machine learning algorithms can essentially be divided into Supervised learning, Semi-supervised learning, Unsupervised learning and Data clustering [2], [3].
All learning algorithms perform model selection and parameter estimation based on one or multiple criteria; in such a framework numerical optimization plays a significant role [4]. In this thesis we focus on Classification, a supervised learning area based on the separation of sets in finite-dimensional spaces (the Feature ones) by means of appropriate separation surfaces. The most popular approach to classification is the Support Vector Machine (SVM) model, where one looks for a hyperplane separating two given sample sets [5].
Optimization methods that seek sparsity of solutions have recently received considerable attention [6], [7], [8], mainly motivated by the need of tackling Feature Selection problems, defined as "the search for a subset of the original measurements features that provide an optimal tradeoff between probability error and cost of classification" [9]. The Feature selection methods are discussed in [10], [11].
In this thesis, we tackle Feature Selection (FS) in the general setting of sparse optimization, where one is faced to the problem [12]:

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{Minimize}} \quad f(x) + \|x\|_0 \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $\|.\|_0$ is the l_0 pseudo-norm, which counts the number of nonzero components of any vector. Sometimes sparsity of the solution, instead of acting on the objective function, is enforced by introducing a constraint on the l_0 pseudo-norm of the solution, thus defining a cardinality-constrained problem [13], [14], [15], [16].
In many applications, the l_0 pseudo-norm in (1.1) is replaced by the l_1-norm, which is definitely more tractable from the computational point of view, yet ensuring sparsity, to a certain extent [17].
In the seminal paper [18], a class of polyhedral norms (the k-norms), inter-

mediate between $\|.\|_1$ and $\|.\|_\infty$, is introduced to obtain sparse approximation solutions to systems of linear equations. Some other norms have been used in different applications [19], [20], [21], [22], [23]. The use of other norms to recover sparsity is described in [24]. In more recent years the use of k-norms has received much attention and has led to several proposals for dealing with l_0 pseudo-norm cardinality constrained problem [25], [26], [27], [28]. In this thesis, we use a new way to take into account the k-norm for sparse optimization.

An alternative way to deal with Feature selection is the multi-objective approach discussed in [29]. Multi-objective optimization is a basic process in many fields of science, including mathematics, economics, management, and engineering applications [30]. In most real situations, the decision-maker needs to make tradeoffs between disparate and conflicting design objectives rather than a single one. Having conflicting objectives means that it is not possible to find a feasible solution where all the objectives could reach their individual optimal, but one must find the most satisfactory compromise between the objectives. These compromise solutions, in which none of the objective functions can be improved in value without impairing at least one of the others, are often referred to as Pareto optimal or Pareto efficient [31]. The set of all objective function values at the Pareto and weak Pareto solutions is said to be the Pareto front (or efficient set) of the multi-objective optimization problem (MOP) in the objective value space [32]. In general, solving a MOP is associated with the construction of the Pareto frontier. The problem of finding the whole solution set of a MOP is important in applications [33]. Many methods have been proposed to find the Pareto front of the multi-objective optimization problems (See [34], [35], [36], [37], [38], [39], [40]).

In this thesis, we have specifically emphasized the application of sparse optimization in Feature Selection for SVM classification. We propose a novel model for sparse optimization based on the polyhedral k-norm. Also, to demonstrate the advantages of considering SVM classification models as multi-objective optimization problems, we propose some multi-objective reformulation of these models. In these cases, a set of Pareto optimal solutions is obtained instead of one in the single-objective cases.

Therefore, our main contribution in this thesis is of two levels: first, propose a new model for sparse optimization based on the polyhedral k-norm for SVM classification, and second, use multi-objective optimization to consider this new model.

The rest of the thesis is organized as follows. Chapter 2 contains some basic concepts and notations about Feature Selection, Binary Classification and the Support Vector Machine. Chapter 3 contains Sparse Optimization via some norms. In Chapter 4, some basic concepts and notations of Multi-Objective Optimization Problems (MOPs) are given. Our approach to sparse optimization via k-norms is presented in Chapter 5, together with a discussion on possible relaxation and algorithmic treatment, and then a reformulation of

the feature selection model in the form of MOPs is given. Also, the results of some numerical experiments on benchmark datasets are in Chapter 5.

2

Feature Selection

In machine learning, providing a pre-process to get better outcomes is necessary. Vast data are collected to train our model and help it learn better. However, generally, the dataset consists of irrelevant data, noisy data and some part of useful data [43]. During machine learning model development, maybe only a few variables in the dataset contribute to model construction, and the remaining features are either redundant or irrelevant. Therefore, it is necessary to identify and select the most appropriate features from the data and remove irrelevant or unimportant ones. This process helps Feature Selection, which is one of the basic concepts of machine learning [44].
In this chapter, the basic concepts of Feature Selection are explained, and then some important and basic models are reviewed.

2.1 Feature Selection Definition

A feature is a characteristic that affects or is useful for a problem, and the selection of essential features for the model is known as Feature Selection. Feature Selection is the process of manually or automatically selecting a subset of the most consistent, non-redundant, and relevant features from the original feature set by removing redundant, irrelevant, or noisy features for use in model building.
Some of the main benefits of performing Feature Selection include the following [80] [41]:

- Simpler models: Simple models are easy to explain. Models that are too complex and inexplicable are not valuable. Feature Selection helps simplify the model so researchers can easily interpret it.
- A more accurate or precise subset of features can reduce the time required to train a model. Feature Selection can reduce training time.
- Overfitting reduction: The accuracy of the estimates obtained for a given simulation can be increased by reducing the variance.

Figure 2.1 can help us see how choosing the best features can help the model perform better by reducing the model input variables and getting rid of noise in the data [46], [41].

As another example, we want to build a model that automatically decides

Fig. 2.1. Feature Selection (Selecting the best features helps the model to perform well) [41].

which old cars should be scrapped. For this purpose, we have a data set that includes the car Model, Year of manufacture, Owner's name and Miles. Consider a table that contains information about these old cars (Figure 2.2). The model has to decide which cars should be scrapped [47], [41].

We understand that the model, year and miles are critical in determining

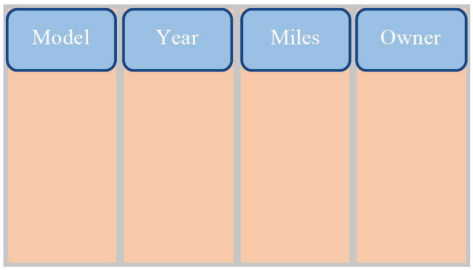

Fig. 2.2. Dataset of old cars [41].

whether the car should be scrapped, but the owner's name cannot be the deciding factor. Furthermore, this extra information can confuse the algorithm in finding patterns between features. Therefore, we can remove this column and select the rest of the features (columns) for the model building (Figure 2.3) [41].

2.1.1 Structure of the Learning System

In terms of label availability, feature selection methods can be classified as follows [48], [42]:

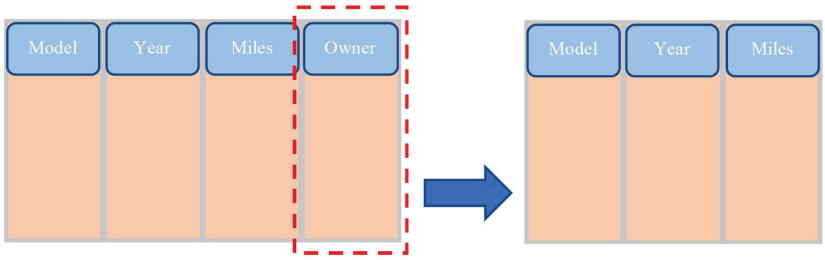

Fig. 2.3. Dropping columns for feature selection [41].

- Supervised,
- Unsupervised, and
- Semi-supervised methods.

And, in terms of different selection strategies, Feature Selection can be categorized as follows [49], [42]:

- Filter,
- Wrapper, and
- Embedded model.

Figure 2.4 shows the classification of Feature Selection methods.

Supervised Feature Selection: Supervised Feature Selection techniques

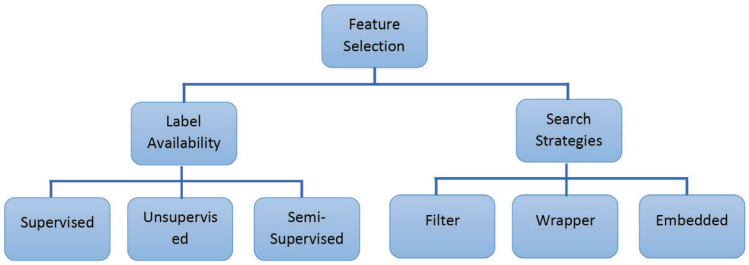

Fig. 2.4. Feature Selection Categories [42].

can be used for the labelled dataset and are usually used for classification tasks. The availability of class labels allows supervised Feature Selection algorithms to select distinctive features to effectively distinguish samples from different classes. A general framework of supervised Feature Selection is shown in Figure 2.5. Features are first generated from the training data and then instead of using all the data to train the supervised learning model, supervised Feature Selection first selects a subset of the features and then combines the data with the selected features. The final selected features and the label information are used to train a classifier, which can be used for the prediction

[59], [42].

Unsupervised Feature Selection: Unsupervised Feature Selection tech-

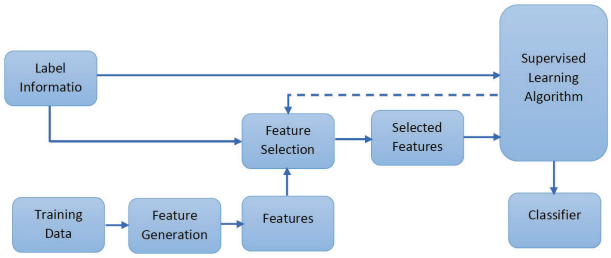

Fig. 2.5. A General Framework of Supervised Feature Selection [42].

niques can be used for the unlabeled dataset and are usually used for clustering tasks. A general framework of unsupervised Feature Selection is described in Figure 2.6. This framework is very similar to supervised Feature Selection, but here no label information is involved in the Feature Selection and model learning steps. Unsupervised Feature Selection relies on alternative criteria without label information to define feature relevance during the Feature Selection phase. One commonly used method is to seek cluster indicators through clustering algorithms and then transform the unsupervised Feature Selection into a supervised framework. [51], [42], [52].

Semi-supervised Feature Selection: When a small portion of the data is

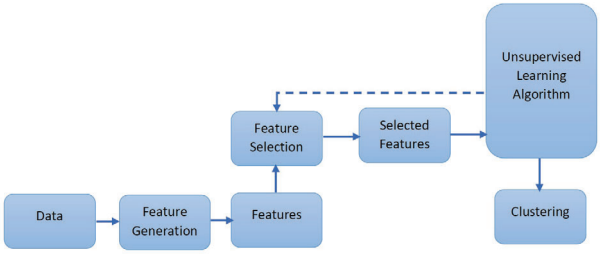

Fig. 2.6. A General Framework of Unsupervised Feature Selection [42].

labeled, Semi-supervised Feature Selection is usually used. When given such data, there may be better choices than selecting a supervised or unsupervised feature. Supervised Feature Selection may fail to select relevant features because the number of labeled data may need to be increased to represent the feature distribution. On the other hand, Unsupervised Feature Selection does not use any labeled data, while the labeled data can provide some discriminating information to select relevant features. Selecting a Semi-supervised

feature, which uses labeled and unlabeled data, is a better choice for handling partially labeled data. The general framework of Semi-supervised Feature Selection is the same as Supervised Feature Selection, except that, in this case, the data is partially labeled. Many existing semi-supervised Feature Selection algorithms are based on constructing the similarity matrix and selecting the features that best fit the similarity matrix. In constructing the similarity matrix, labeled and unlabeled data are used. Label data can provide discriminative information to select relevant features, while unlabeled data provide complementary information [52], [42].

Filter Models: In Filter Method, features are selected based on statistical measures. The filter method does not depend on the learning algorithm and selects features as a pre-processing step. This method filters out the model's irrelevant features and extra columns by using different measures. For filter models, features are selected based on the characteristics of the data without utilizing learning algorithms. A filtering algorithm usually consists of two steps. In the first step, features are ranked based on specific criteria. In the second step, features with the highest rankings are chosen [53], [42].

Wrapper Models: In the Wrapper algorithm, Feature Selection is done by treating it as a search problem where different combinations are constructed and evaluated and also compared with other combinations. This algorithm is trained iteratively using a subset of features. In the filter approach, the optimal feature subset depends on the specific biases and heuristics of the learning algorithms. Based on this assumption, wrapper models use a specific learning algorithm to evaluate the quality of the selected features. A general framework of the wrapper model is shown in Figure 2.7 [54], [55], [42].

Embedded Models: Embedded methods combine the advantages of both

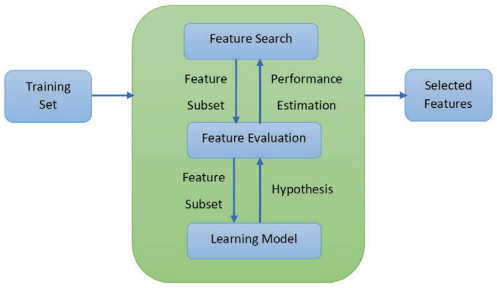

Fig. 2.7. A General Framework of Wrapper Model [42].

filter and wrapper methods by considering feature interactions and low computational costs. These methods have fast processing similar to the filter method, but they are more accurate than the filter method. Embedded models are a trade-off between the filter and wrapper methods by embedding the Feature Selection into the model construction. Thus, embedded models take advantage

of both filter and wrapper models: (1) they are less computationally intensive than wrapper methods because these models do not need to run the learning models several times to evaluate the features, and (2) they include the interaction with the learning model. These methods are iterative and also evaluate each iteration and find the important features in a particular iteration [56], [42].

2.1.2 How to choose a Feature Selection Method?

How to choose a Feature Selection Method? In machine learning, it is essential to determine which feature selection method works appropriately for the model. The more the data types of the variables are known, the easier it will be to choose the appropriate statistical criteria for Feature Selection. Therefore, the types of input and output variables must be identified first. In machine learning, variables are of mainly two types [57], [58], [59], [42]:

- Numerical Variables: Variables with continuous values such as integer, and float.
- Categorical Variables: Variables with categorical values such as Boolean, ordinal, and nominals.

2.2 Basic Concepts and Notations

This section presents some basic concepts and notations that make more accessible the understanding of this thesis. We start by giving a brief description of the classification problem (especially binary classification) in supervised learning. We then focus on a specific task: support vector machine in feature selection.

2.2.1 Classification Problem

We consider a classification problem which is formulated in the following way: Suppose that there is a set of objects, perhaps infinite (observations, patterns, Etc.), which can be classified into two classes of data (that is, assigned to two sets). We want to define an algorithm that, with the minimum error, will classify objects from the entire set [60].
In this thesis, we have focused on binary classification, but there are also methods for classifying more than two classes. Classification problems with more than two class labels are known as multi-class classification. Each entity is assigned to one class without overlap, and each sample can only be labelled as one class. For example, consider a classification using extracted features that includes a set of images of three types of fruits (oranges, apples, or pears). Each image is one sample labelled as one of the three possible

classes. This assumption makes in multi-class classification that each sample is assigned to one and only one label (for example, one sample cannot be both a pear and an apple).
In a classification, objects are represented by vectors in the vector space V. Although SVMs can be used on any arbitrary vector spaces, the vector space V is simply space $\mathbb{R}^n$. In this space, vector x is a set of n real numbers x_i (components of the vector) $x = (x_1, \ldots, x_n)$ [61].
In the rest of this thesis, all of the concepts are used in the terms of binary classification.

Definition 2.1. *(**Training Set**) [62], [63] A sample of objects with known class labels is called a training set and is written as:*

$$(x_1, y_1), \ldots, (x_m, y_m) \tag{2.1}$$

where $y_i \in \{\pm 1\}$ is the class label of vector x_i, and m is the size of the training set.

Definition 2.2. *(**Decision Function**) [62], [63] A classification algorithm (classifier) is represented with a decision function as follows:*

$$f : \mathbb{R}^n \to \{\pm 1\} \tag{2.2}$$

such that $f(x) = +1$ if the classifier assigns x to the first class, and $f(x) = -1$ if the classifier assigns x to the second class.

2.2.2 Equation of a Hyperplane

In $\mathbb{R}^n$ space the following equation defines a $(n-1)$-dimensional set of vectors called hyperplane:

$$wx + b = \sum_{i=1}^{n} w_i x_i + b = 0 \tag{2.3}$$

That is, for a given nonzero vector $w = (w_1, \ldots, w_n) \in \mathbb{R}^n$ and a scalar $b \in \mathbb{R}$, the set of all vectors $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ satisfying equation (2.3) forms a hyperplane.
In the rest of this thesis, we will denote a hyperplane by letter π or by $\pi(w; b)$. The term "hyperplane" means that the dimensions of the plane are one size smaller than the dimensions of the entire space $\mathbb{R}^n$. For example, a point is a hyperplane in $\mathbb{R}$; a line is a hyperplane in $\mathbb{R}^2$ (Figure 2.8); a plane is a hyperplane in $\mathbb{R}^3$; a three-dimensional space is a hyperplane in $\mathbb{R}^4$, and so on [62], [63], [64].

Vector w is called the normal vector of the hyperplane, and b is called the intercept of the hyperplane $\pi(w; b)$. The normal vector defines the orientation of the hyperplane in space, while the ratio between normal vector

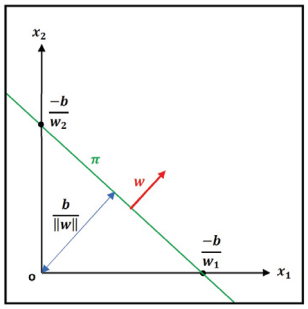

Fig. 2.8. Hyperplane π in two-dimensional space ($\mathbb{R}^2$) is a line. [63].

and intercept of the hyperplane ($\|w\|$ and b) defines the distance between the hyperplane and the origin of space (The norm adopted here is the Euclidean one). The normal vector w is perpendicular to all vectors parallel to the hyperplane [62], [63], [64].

Definition 2.3. ***(Half-Spaces)*** *[62], [63] Hyperplane π divides coordinate space $\mathbb{R}^n$ into two parts located sidewise of the hyperplane, called positive and negative half-spaces. The positive half-space is pointed by the normal vector of the hyperplane. For any vector x in positive half-space we have $wx + b > 0$, while for any vector x in negative half-space we have $wx + b < 0$ (See Figure 2.9).*

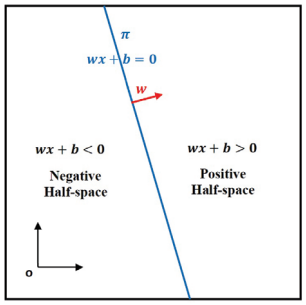

Fig. 2.9. Negative and positive half-spaces are defined by hyperplane π [63].

2.2.3 How w and b define the position of the hyperplane?

(**1.**) The origin of space is in the positive half-space of the hyperplane $\pi(w, b)$ if $b > 0$, and in the negative half-space if $b < 0$. If $b = 0$ then the hyperplane passes through the origin (See Figure 2.10 (a)).

(**2.**) We can move the hyperplane parallel to itself in the direction from the origin by increasing the absolute value $\mid b \mid$ of the intercept. And also we can move the hyperplane towards the origin by decreasing $\mid b \mid$ (See Figure 2.10 (a)).
(**3.**) We can move the hyperplane in a circle around the origin by changing the normal vector w in a way that preserves its norm (the radius of the circle is $\frac{\mid b \mid}{\|w\|}$) (Figure 2.10 (b)).
(**4.**) We can move the hyperplane parallel to itself from the origin by reducing the length of the normal vector w in a way that preserves its direction. We can move the hyperplane towards the origin by increasing the length of the normal vector w in a way that preserves its direction (Figure 2.10 (c)). Thus, a hyperplane can be moved parallel to itself not only by changing intercept b, but also by scaling normal vector w [62], [63].

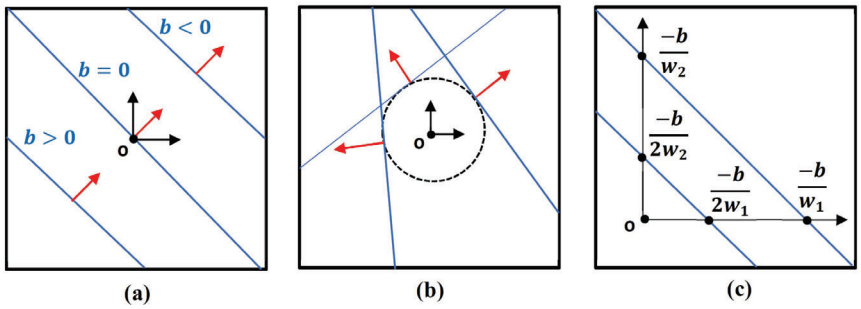

Fig. 2.10. Understanding the meaning of hyperplane parameters w and b [63].

2.2.4 Hyperplane for Separating two Classes of Data

Hyperplane $\pi(w, b)$, separates two classes (sets) of vectors (C_1 (Class 1) and C_2 (Class 2)) if either [62], [63], [64], [65]:

$$\begin{aligned} wx + b > 0, \quad \forall x \in C_1 \\ wx + b < 0, \quad \forall x \in C_2 \end{aligned} \tag{2.4}$$

or

$$\begin{aligned} wx + b < 0, \quad \forall x \in C_1 \\ wx + b > 0, \quad \forall x \in C_2 \end{aligned} \tag{2.5}$$

Definition 2.4. *(**Linearly Separable**) [62], [63], [64], [65] Two classes (sets) of data are called linearly separable if there exists at least one hyperplane that separates them. If hyperplane $\pi(w, b)$ separates classes C_1 and C_2*

according to (2.4) or (2.5) the following decision function gives us a classifier that correctly classifies all vectors from two classes C_1 and C_2:

$$f(x) = sgn\{wx + b\} = \begin{cases} +1, if\ wx + b \geq 0 \\ -1, if\ wx + b < 0 \end{cases}$$

Definition 2.5. *(**Distance Between Vector and Hyperplane**) [62], [63] The distance $d(x;\pi)$ between vector x and hyperplane $\pi(w,b)$ can be calculated according to the following equation:*

$$d(x;\pi) = \frac{wx + b}{\|w\|} \tag{2.6}$$

In equation (2.6):
$d(x;\pi) > 0$ when x is in positive half-space,
$d(x;\pi) < 0$ when x is in negative half-space, and
$d(x;\pi) = 0$ when x is placed on the hyperplane π.
In equation (2.6) for the distance, if $\|w\| = 1$, then is simply $d(x;\pi) = wx + b$. Also, it follows from equation (2.6) that the distance between the origin of space and hyperplane π is equal to $\frac{b}{\|w\|}$. This fact allows us to make several useful observations regarding the position and orientation of the hyperplane in space, and how parameters b and w affect them [62], [63], [64], [65].
The margin of two separating classes C_1 and C_2 by hyperplane π is denoted by $m(\pi, C_1, C_2)$ and is defined as the distance between π and class C_1, plus the distance between π and class C_2 (see Figure 2.11 (a)) [62], [63], [64], [65]:

$$m(\pi, C_1, C_2) = d(\pi; C_1) + d(\pi; C_2) \tag{2.7}$$

The distance between hyperplane π and a set of vectors C is defined as the minimum distance between π and vectors from C:

$$d(\pi; C) = min_{x \in C}\ |d(x;\pi)| \tag{2.8}$$

In this definition the absolute value of the signed distance $d(x;\pi)$ defined by equation (2.6) are using.
Equivalently, the margin can be defined as the distance between classes C_1 and C_2 measured along the normal vector w (see Figure 2.11 (b)). If C_1^w is the set containing projections of all vectors from C_1 onto the line parallel to vector w, and C_2^w is the set containing similar projections of all vectors from C_2, then the margin of two separating classes is defined as follows [62], [63], [64], [65]:

$$m(\pi, C_1, C_2) = d(C_1^w; C_2^w), \tag{2.9}$$

where

$$d(C_1^w; C_2^w) = min_{x_1 \in C_1^w, x_2 \in C_2^w}\ d(x_1; x_2) \tag{2.10}$$

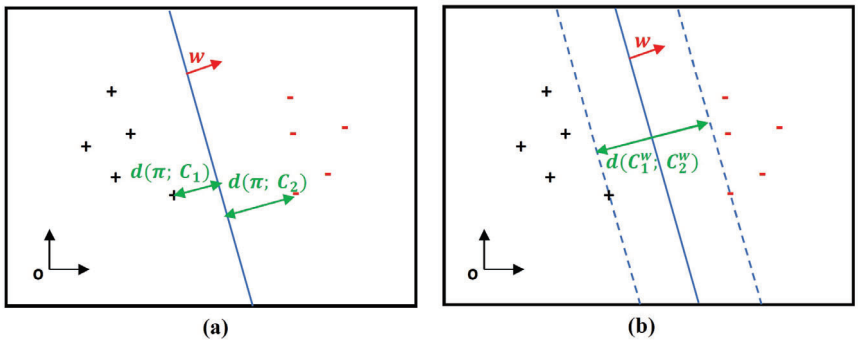

Fig. 2.11. (a) Margin of hyperplane π is the distance from π to the first class (minus symbol) plus the distance from π to the second class (plus symbol). (b) Equivalently, it can be defined as the distance between two classes measured along the normal vector w of the hyperplane [63].

2.2.5 Hyperplane for Separating Two Classes of Data

It is clear that for two linearly separable classes of data, there always exists an infinite number of hyperplanes (with differently oriented w and different b) that separate them. However, the critical question is which of these hyperplanes is better and should be used to define a classifier? (see Figure 2.12) [62], [63]

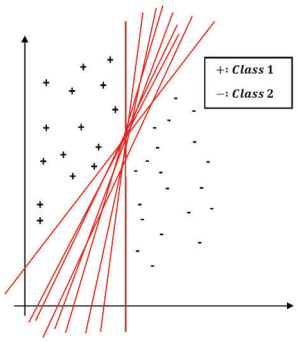

Fig. 2.12. Many hyperplanes can be fit to classify two sets of data, but which one is the best? [63].

We need to find a linear classifier that achieves maximum separation, but only one of many linear classifiers (hyperplanes) that separates the two data sets achieves maximum separation. The Support Vector Machine (SVM) chooses one of the hyperplanes with the maximum margin. We need the hy-

perplane with the maximum margin because if we use a hyperplane to classify, it might end up closer to one set of data than others, and we do not want this to happen [66], [67].
Suppose that there are two classes of linearly separable training vectors, the support vector machine defined on such a training set is a classifier where $\pi = wx + b$ is the equation of the hyperplane that separates two classes by the maximum margin and is equidistant from both classes (see Figure 2.13) [62], [63], [64], [65], [66].
Parameters w, b of the SVM hyperplane can be found as a solution to the

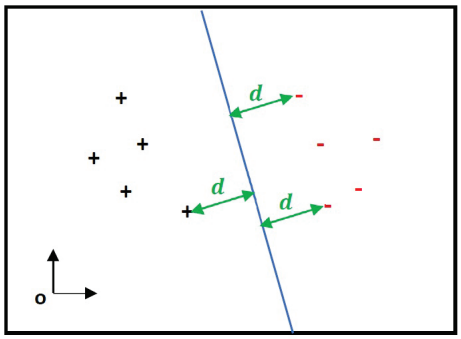

Fig. 2.13. Maximum margin hyperplane for two linearly separable classes: $d = d(\pi; C_1) = d(\pi; C_2)$ is maximized [63].

following optimization problem [62], [63]:

$$\begin{aligned} &\underset{\mathbf{w},b}{\text{Minimize}} \ \frac{1}{2}\|w\|^2 \\ &\text{subject to} \\ &\qquad wx + b \geq 1, \quad \forall x \in C_1 \\ &\qquad wx + b \leq -1, \ \ \forall x \in C_2 \end{aligned} \tag{2.11}$$

where C_1 and C_2 are two classes of training data. In problem (2.11), parameter b is the optimization variables, but it is not present in the objective function. The optimization problem (2.11) has a quadratic objective function and linear constraints. Thus it is a quadratic programming problem. The properties of quadratic programming problems are well known, and there are very efficient algorithms for solving these types of problems [62], [63], [64], [65], [66].
The objective function of problem (2.11) is strictly convex (since the matrix of its second-order derivatives -the Hessian- is positive definite), and the feasible region defined by linear inequalities is also convex. Therefore, this problem will have a unique solution (global minimum) (w^*, b^*).
The feasible region defined by the constraints of problem (2.11) will be empty, and the problem will have no feasible solution: If two classes are not linearly

separable and also if the training set contains only one class.
Also, an important question arises: Why can we find the hyperplane parameters with the maximum margin by solving the problem (2.11)? To answer this question, this problem can be transformed into an equivalent problem that has a clearer geometric interpretation.
First, minimizing $\frac{1}{2}\|w\|^2$ is equivalent to minimizing $\|w\|$, which in turn is equivalent to maximizing $\frac{1}{\|w\|}$, so we can rewrite problem (2.11) as following problem [62], [63], [64], [65], [66]:

$$\begin{aligned} &\underset{\mathbf{w},b}{\text{Maximize}} \ \frac{1}{\|w\|} \\ &\text{subject to} \\ &\qquad wx+b \geq 1, \quad \forall x \in C_1 \\ &\qquad wx+b \leq -1, \ \forall x \in C_2 \end{aligned} \tag{2.12}$$

Second, we can divide constraints of problem (2.12) by a positive number $\|w\|$ [62], [63], [64], [65], [66]:

$$\begin{aligned} &\underset{\mathbf{w},b}{\text{Maximize}} \ \frac{1}{\|w\|} \\ &\text{subject to} \\ &\qquad \frac{wx+b}{\|w\|} \geq \frac{1}{\|w\|}, \quad \forall x \in C_1 \\ &\qquad \frac{wx+b}{\|w\|} \leq \frac{-1}{\|w\|}, \quad \forall x \in C_2 \end{aligned} \tag{2.13}$$

On the other hand, according to equation (2.6), $\frac{wx+b}{\|w\|}$ is the distance ($d(x;\pi)$) between hyperplane $\pi(w;b)$ and point x , and with introducing new variable $q = \frac{1}{\|w\|}$, the following problem, which is equivalent to the problem (2.11), is obtained [62], [63], [64], [65], [66]:

$$\begin{aligned} &\underset{\mathbf{w},b}{\text{Maximize}} \ q \\ &\text{subject to} \\ &\qquad d(x;\pi) \geq q, \quad \forall x \in C_1 \\ &\qquad d(x;\pi) \leq -q, \quad \forall x \in C_2 \end{aligned} \tag{2.14}$$

That is, it finds the parameters w and b so that maximize the margin $m = 2q$ between π , C_1 and C_2 .
The connection between parameters w, b and q can be shown geometrically as follows:
Suppose that draw a spherical hull of radius $q = \frac{1}{\|w\|}$ around each training point x. Consider some feasible hyperplane $\pi(w;b)$. According to the constraints of problem (2.14), this hyperplane must separate our points together

with their hulls (see Figure 2.14 (a)). Now suppose that we want to increase q twofold. Since q is a function of $\|w\|$, we have to decrease $\|w\|$ twofold. If we divide vector w by two, we move our hyperplane parallel to itself further from the origin. However, if we divide by two vectors w and intercept b, we do not move the hyperplane. This way, downscaling w and b, we increase the radius of the hulls while keeping hyperplane π in the same position and orientation, until at least one hull touches it (see Figure 2.14 (b)) [62], [63], [64], [65], [66]. If we have space to move parallel to itself away from the hull that touches it, we can do it by changing b only, and letting the hulls grow further. At some point, our hulls will reach the maximum size q achievable for hyperplanes with normal vectors collinear to w (see Figure 2.14 (c)). If we have space for the hulls to grow further, we can change the orientation of π by changing components of vector w, while keeping $\|w\|$ equal to the current value of $\frac{1}{q}$. Doing so and adjusting b, we keep π feasible and increase q until we arrive at the optimal configuration (see Figure 2.14 (d)) [62], [63], [64], [65], [66].

Therefore, the aim of a support vector machine (SVM) is to orient this hy-

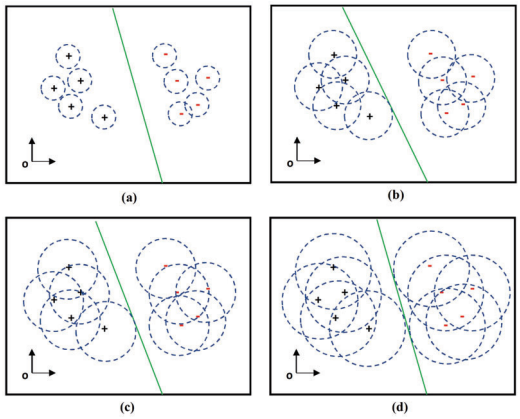

Fig. 2.14. Finding maximum margin hyperplane: geometrical insight [63].

perplane in such a way that it has as far as possible distance from the closest members of both classes (maximum margin) while being equally distant from both classes (see Figure 2.15) [62], [63], [64], [65], [66].

2.2.6 Binary Classification for Data that is not Fully Linearly Separable

In order for the SVM method to be applied to data that is not fully linearly separable, the constraints of problem (2.11) are slightly reduced (relaxed) to allow for misclassified points. This is done by introducing a positive slack variable ξ_i, $i = 1, \ldots, L$ in problem (2.11) [62], [63], [64], [65], [66]:

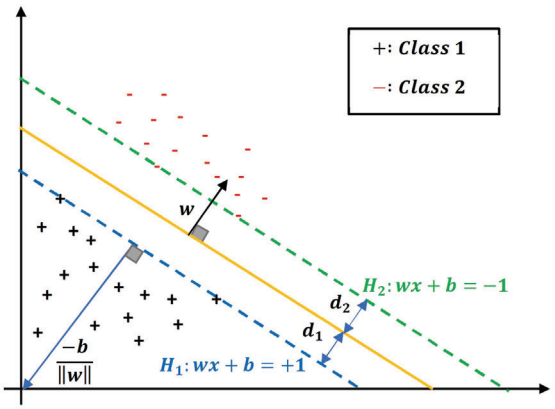

Fig. 2.15. Hyperplane with maximum margin for two linearly separable classes [62], [63].

$$\begin{aligned} wx_i + b &\geq +1 - \xi_i, \quad && \forall x_i \in C_1 \\ wx_i + b &\leq -1 - \xi_i, \quad && \forall x_i \in C_2 \\ \xi_i &\geq 0, && \forall i \end{aligned} \tag{2.15}$$

Which can be combined into:

$$y_i(wx_i + b) - 1 + \xi_i \geq 0, \quad where \quad \xi_i \geq 0, \quad \forall i \tag{2.16}$$

In this soft margin SVM, a penalty is applied to the data points on the incorrect side of the margin boundary. The value of this penalty increases with the distance from the margin boundary. Since soft margin SVM aims to reduce the number of misclassifications, an appropriate way is the following optimization problem [62], [63], [64], [65], [66]:

$$\begin{aligned} &\underset{\mathbf{w},b}{\text{Minimize}} \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{L}\xi_i \\ &\text{subject to} \\ &\qquad y_i(wx_i + b) - 1 + \xi_i \geq 0, \quad \forall i \\ &\qquad \xi_i \geq 0, \quad \forall i \end{aligned} \tag{2.17}$$

Where parameter C controls the trade-off between the slack variable penalty and the size of the margin [62], [63], [64], [65], [66].

2.2.7 The Effect of Parameter C

Positive constant parameter C in the objective function of the Soft margin SVM problem (2.17) should be adjusted by the user. This parameter balances

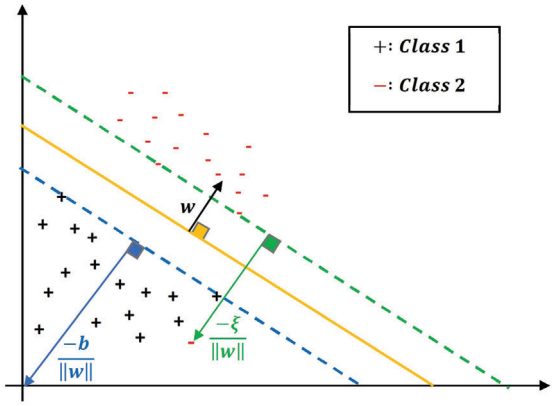

Fig. 2.16. Hyperplane through two non-linearly separable classes [62], [63].

two goals: maximizing the margin and minimizing the number of misclassification (errors value) on the training data. These goals may be conflicting since margin expansion may increase the error value (see Figure 2.17) [62], [63], [64], [65], [66].

We can choose to favour one goal over another by changing parameter C. This

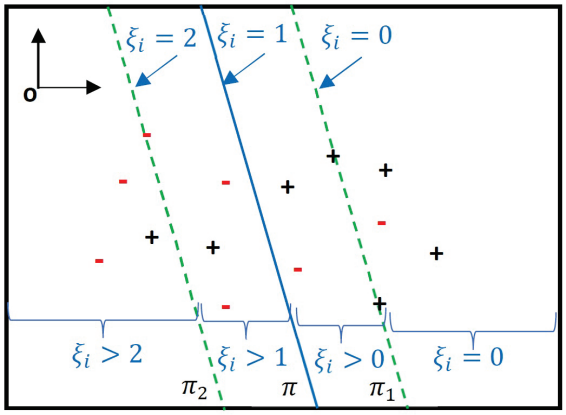

Fig. 2.17. The values of slack variables ξ_i showed for training points of Class 2 (Negative symbol). Optimal hyperplane π is defined by $wx + b = 0$, π_1 is defined by $wx + b = 1$, and π_2 is defined by $wx + b = -1$. The margin is the region between hyperplane π_1 and π_2. Expanding the margin may increase the overall error. It should also be noted that points located in the margin will always have $\pi > 0$, even when they are correctly classified by the hyperplane [63].

is illustrated in Figure 2.18. In this figure, four separating hyperplanes and

their margins are shown, which were obtained for the same training data set using increasing values of the parameter C. Circled are points with non-zero error terms ξ_i. These points lie either on the hyperplane's wrong side or in the margin. When parameter C is chosen very small, the sum of error terms becomes negligible in the objective function of the problem (2.17), so the goal of optimization is to maximize the margin. In this case, the margin can be large enough to contain all points. At another extreme, when parameter C is chosen very large, the sum of error terms dominates the margin term in the objective function of the problem (2.17). So the goal of optimization is to minimize the sum of error terms. In this case, the margin can be so small that it does not contain any points. However, it should be noted in Figure 2.18 that despite the difference in the value of parameter C and the size of the margin, all four hyperplanes shown are fairly similar, and they all correctly classify the same training data points [62], [63], [64], [65], [66], [68].

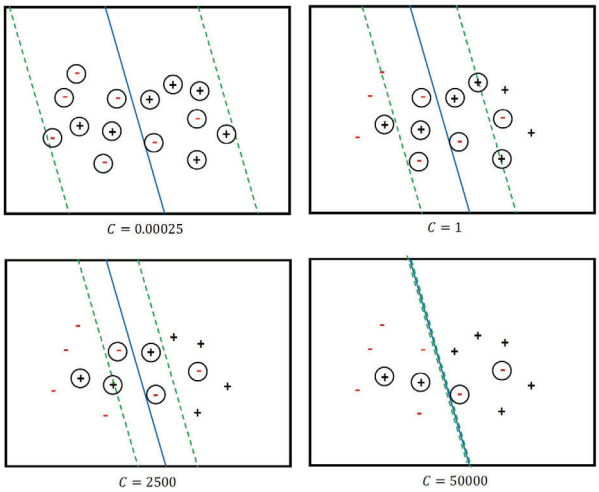

Fig. 2.18. Optimal separating hyperplanes and their margins obtained for the same training set by using different values of parameter C [63].

2.2.8 Binary Classification

In this thesis we consider the classification task in the basic form of binary classification. In binary classification we have the representation of two classes of individuals in the form of two finite sets $\mathcal{A}$ and $\mathcal{B} \subset \mathbb{R}^n$, such that $\mathcal{A} \cap \mathcal{B} = \emptyset$, and we want to classify an input vector $x \in \mathbb{R}^n$ as a member of the class represented by $\mathcal{A}$ or that by $\mathcal{B}$. The training set for binary classification is defined as follows [69]:

$$T = \{(x_i, y_i) : x_i \in \mathbb{R}^n, y_i \in \{\pm 1\} \quad and \quad i = 1, \ldots, m\} \tag{2.18}$$

with the two sets $\mathcal{A}$ and $\mathcal{B}$ labelled by $+1$ and -1, respectively. The functional dependency $f : \mathbb{R}^n \to \{\pm 1\}$, which determines the class membership of a given vector x, assumes the following form [69], [70], [71]:

$$f(x) = \begin{cases} +1, if\ x \in \mathcal{A} \\ -1, if\ x \in \mathcal{B} \end{cases}$$

Assume that the two finite point sets $\mathcal{A}$ and $\mathcal{B}$ in $\mathbb{R}^n$ consist of m and k points respectively. They are associated to the matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{k \times n}$, where each point of a set is represented as a row of the corresponding matrix. In the classic SVM method we want to construct a separating hyperplane:

$$P = \left\{x : x \in \mathbb{R}^n, \quad x^T w = \gamma\right\} \tag{2.19}$$

with normal $w \in \mathbb{R}^n$ and distance [69]:

$$\frac{|\ \gamma\ |}{\|w\|_2} \tag{2.20}$$

to the origin. The separating plane P determines two open halfspaces:

- $P_1 = \left\{x : x \in \mathbb{R}^n, \quad x^T w > \gamma\right\}$ it is intended to contain most of the points belonging to $\mathcal{A}$.
- $P_2 = \left\{x : x \in \mathbb{R}^n, \quad x^T w < \gamma\right\}$ it is intended to contain most of the points belonging to $\mathcal{B}$.

Therefore, letting e be a vector of ones of appropriate dimension, we want to satisfy the following inequalities:

$$Aw > e\gamma, \quad Bw < e\gamma \tag{2.21}$$

to the possible extent. The problem can be equivalently put in the form.

$$Aw \geq e\gamma + e, \quad Bw \leq e\gamma - e \tag{2.22}$$

Conditions (2.21) and (2.22) are satisfied if and only if the convex hulls of $\mathcal{A}$ and $\mathcal{B}$ are disjoint (the two sets are linearly separable) [69].
Application of Feature Selection to SVM, as we will see next, amounts to suppressing as many of the components of w as possible.

2.2.9 Support Vector Machine and Feature Selection

In real-world classification problems based on supervised learning, the information available are the vectors a_i's and b_l's (the rows of A and B, respectively) defining the (labelled) training set. [72], [73]. The standard formulation of SVM is the following, where variables y_i and z_l represent the classification error associated to the points of $\mathcal{A}$ and $\mathcal{B}$, respectively:

$$\begin{aligned} \text{Min} \quad & C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) + \|w\|_2^2 \\ & \text{subject to} \\ & -a_i^T w + \gamma + 1 \leq y_i, \quad i = 1, \ldots, m_1 \\ & b_l^T w - \gamma + 1 \leq z_l, \quad l = 1, \ldots, m_2 \\ & y_i, z_l \geq 0. \end{aligned} \tag{2.23}$$

Positive parameter C defines the trade-off between the objectives of minimizing the classification error and maximizing the separation margin.
By replacing l_2 with l_1 in model (2.23), we will have the following model:

$$\begin{aligned} \text{Min} \quad & C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) + \|w\|_1 \\ & \text{subject to} \\ & -a_i^T w + \gamma + 1 \leq y_i, \quad i = 1, \ldots, m_1 \\ & b_l^T w - \gamma + 1 \leq z_l, \quad l = 1, \ldots, m_2 \\ & y_i, z_l \geq 0. \end{aligned} \tag{2.24}$$

Feature selection is primarily performed to select informative features [69], and has become one of the most important issues in the field of machine learning [74].

Referring to the above model, the goal is to construct a separating plane that gives good performance on the training set while using a minimum number of problem features. This objective can be pursued by a looking for a normal w to the separating hyperplane characterized by the smallest possible number of non-zero components. This can be achieved by adding a sparsity enforcing term to the objective function [69], [74].
As we will see next, a companion model aimed at suppressing as many elements of w as possible, known as LASSO approach, is obtained by replacing l_2-norm with the l_1 [69], [73].

3

Sparse Optimization

In this chapter, we will first define the several types of norms and then investigate the sparsity optimization problems through norms. In particular, we will focus on l_0-pseudo-norm and polyhedral k-norm.

3.1 *Lp* Norm

In this section, the definition of different norms is discussed.

Definition 3.1. ***(Norm of a Vector)*** *[75] Given a vector space V, a norm is a function $\|.\| : V \to \mathbb{R}_+$ that assigns a non-negative real value (length) to each vector in V, and has the following properties [75]:*
1. $\|x\| \geq 0$.
2. $\|x\| = 0$ *if and only if* $x = 0$. *This means that the norm of a non-zero vector is non-zero.*
3. $\|x+y\| \leq \|x\| + \|y\|$. *This means the norm of a vector sum does not exceed the sum of the norms of its parts (the triangle inequity).*
4. $\|\alpha x\| = |\alpha| \, \|x\|$. *This means scaling a vector scales its norm by the same amount.*

A vector space with a norm is called a normed vector space. The norm of a vector w is denoted as $\|w\|$. For all vectors x and y, for all scalars $\alpha \in \mathbb{R}$, a normed vector space satisfies the conditions 1-4 to conform to a reasonable notion of length.
The Lp norm of $x \in \mathbb{R}^n$ is defined by [75]:

$$\|x\|_p = (|x_1|^p + |x_2|^p + \ldots + |x_n|^p)^{\frac{1}{p}} \tag{3.1}$$

For $p = 0$ in relation (3.1), $\|x\|_0$ is defined as follows [75]:

$$\|x\|_0 = \lim_{p \to 0} (|x_1|^p + |x_2|^p + \ldots + |x_n|^p) \tag{3.2}$$

This actually is the number of non-zero entries of the vector x. The l_0-Pseudo-norm of the vector x, $\|x\|_0$, is also called the support or the cardinality of x. For any vector $x \in \mathbb{R}^n$, we define its pseudo-norm $l_0(x)$ by:

$$l_0(x) = \|x\|_0 = number\ of\ nonzero\ components\ of\ x. \tag{3.3}$$

The function pseudo-norm $l_0 : \mathbb{R}^n \to \{0, 1, \ldots, n\}$, satisfies 3 out of 4 axioms of a norm (Definition 3.1) [75]:

- We have: $\|x\|_0 \geq 0$. .
- We have: $\|x\|_0 = 0$ if and only if $x = 0$.
- We have: $\|x + y\|_0 \leq \|x\|_0 + \|y\|_0$.
- But 0-homogeneity holds true: $\|\alpha x\|_0 = \|x\|_0, \forall \alpha \neq 0$.

3.2 Sparsity Inducing *Lp* Norms

The purpose of the sparse SVM method is to control the number of non-zero components of the normal vector to the separating hyperplane while maintaining satisfactory classification accuracy [14]. Therefore, the following two objectives should be minimized [69]:

- The number of misclassified training data;
- The number of nonzero elements of vector w (The normal vector of the separating hyperplane).

We consider sparse optimization problem of the following form [76]:

$$\begin{aligned} \underset{w \in \mathbb{R}^n}{\text{Minimize}} \quad & f(w) + \Omega(w) \\ \text{subject to} \quad & \\ & w \in S \end{aligned} \tag{3.4}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex differentiable function and $\Omega : \mathbb{R}^n \to \mathbb{R}$ is a sparsity inducing Lp norm [76].

3.2.1 Sparsity Through l_1-Norm

One of the norms that induce sparsity is the l_1-norm. In the sense that some coefficients of the normal vector of the separating hyperplane (vector w), depending on the strength of the norm, will be equal to zero. In this case, problem (3.4) becomes the following problem [76], [77], [78]:

$$\begin{aligned} \underset{w \in \mathbb{R}^n}{\text{Minimize}} \quad & f(w) + \|w\|_1 \\ \text{subject to} \quad & \\ & w \in S \end{aligned} \tag{3.5}$$

3.2.2 Sparsity Through l_0-Pseudo-Norm

The problem of minimizing the l_0-pseudo-norm of the decision variables vector, subject to a number of constraints, has become of great importance. For example in machine learning, $l_0(x)$-pseudo-norm minimization is used for feature selection, minimization of training error and ensuring sparsity in solutions.
In this case, problem (3.4) becomes the following problem [74]:

$$\begin{aligned} \underset{w\in\mathbb{R}^n}{\text{Minimize}} \quad & f(w) + \|w\|_0 \\ \text{subject to} \quad & \\ & w \in S \end{aligned} \tag{3.6}$$

Minimization of the l_0-pseudo-norm provides a natural way of directly addressing the feature selection and pattern classification objectives in a single optimization. However, this is achieved at the cost of having to solve a very difficult optimization problem which will not necessarily generalize well. This combinatorial optimization problem is NP-Hard [79].

Definition 3.2. ***(NP-Hardness)*** *A class of computational problems for which every given yes-solution can be verified as a solution in polynomial time by a deterministic Turing machine (or solvable by a nondeterministic Turing machine in polynomial time), are called NP problems. If an algorithm for solving a problem can be translated into an algorithm for solving any NP problem (nondeterministic polynomial time), then the problem is NP-hard. Problems that are NP-hard need not be NP elements. Indeed, they may not even be decidable [79], [80], [81].*

It is shown in [79] that problem (3.6) with l_0-pseudo-norm, is an NP-hard problem. It cannot even be approximated within $2^{log^{1-\epsilon}(n)}$, for all $\epsilon > 0$ unless $NP \subset DTIME(n^{polylog(n)})$ where $DTIME(x)$ is the class of deterministic algorithms ending in $O(x)$ steps. It means that under rather general assumptions, there is no polynomial time algorithm that can approximate the value of the objective function at optimum N_0 within less than $N_0(2^{log^{1-\epsilon}(n)})$ for all $\epsilon > 0$. Therefore, the minimization of the problem (3.6) is hopeless, and very specific approximations must be defined by well-motivated discussions and experiments [79], [80].
The simplest approach to make the problem (3.6) tractable, can be that of replacing the l_0-pseudo-norm, which is a non-convex discontinuous function, by the l_1-norm, thus obtaining the following linear programming problem [74], [73]:

$$\begin{aligned} \underset{w,y\in\mathbb{R}^n}{\text{Minimize}} \quad & f(w)+\sum_{i=1}^{n} y_i \\ \text{subject to} \quad & \\ & w \in S \\ & -y \leq w \leq y \end{aligned} \tag{3.7}$$

This problem can be solved effectively even when its dimensions are very large. Also, under appropriate assumptions in the polyhedral set P, it can be proved that by solving the problem (3.7), the solution (3.6) can be obtained (see, e.g., [74], [82]). But these assumptions may not be satisfied in many cases. Some experiments concerning machine learning problems and reported in [74] show that a concave optimization based approach performs better than that based on the employment of the l_1-norm.
In order to show the underlying idea of the concave approach, the objective function of problem (3.6) is rewritten as follows [74]:

$$\|w\|_0 = \sum_{i=1}^{n} s(|w_i|) \tag{3.8}$$

where $s : \mathbb{R} \to \mathbb{R}$ is the step function such that $s(t) = 1$ for $t > 0$ and $s(t) = 0$ for $t \leq 0$. The nonlinear approach experimented in [74], [73] was originally proposed in [82], and is based on the idea of replacing the discontinuous step function by a continuously differentiable concave function $v(t) = 1 - e^{-\alpha t}$, with $\alpha > 0$, thus obtaining a problem of the following form [74]:

$$\begin{aligned} \underset{w,y\in\mathbb{R}^n}{\text{Minimize}} \quad & f(w)+\sum_{i=1}^{n} (1-e^{-\alpha y_i}) \\ \text{subject to} \quad & \\ & w \in S \\ & -y \leq w \leq y \end{aligned} \tag{3.9}$$

The replacement of problem (3.6) by the smooth concave problem (3.9) is well-motivated (see [74], [83]) both from a theoretical and a computational point of view:

- For sufficiently large values of the parameter α there exists a solution for problem (3.9) which provides a solution of the original problem (3.6), and in this sense the approximating problem (3.9) is equivalent to the given nonsmooth problem (3.6);
- The Frank-Wolfe algorithm [74], [84] with unitary step-size is guaranteed to converge to a vertex stationary point of the problem (3.9) in a finite number of iterations (this convergence result was proved for a general class of concave programming problems); The algorithm thus requires solving a finite sequence of linear programs to compute a fixed point of the problem

(3.9), and this may be quite advantageous from a computational point of view.

A similar concave optimization-based approach has been proposed in [74], [85], where the main idea is to use the logarithm function instead of the step function, and this leads to the following smooth-concave problem:

$$\begin{aligned} \underset{w,y\in\mathbb{R}^n}{\text{Minimize}} \quad & f(w)+\sum_{i=1}^{n} ln(\epsilon+y_i) \\ \text{subject to} \quad & \\ & w \in S \\ & -y \leq w \leq y \end{aligned} \tag{3.10}$$

with $0 < \epsilon \leq 1$. Formula (3.10) derives from the fact that, due to the form of the logarithmic function, it is better to increase one variable y_i while making another variable zero instead of making a compromise between both variables. And this can facilitate the calculation of a sparse solution [74], [85], and similarly to [74], the Frank-Wolfe algorithm [84] with unitary step-size has been applied to solve (3.10), and good computational results have been obtained.

3.3 Feature Selection in SVM with l_0-Pseudo-Norm

We tackle Feature Selection in SVM as a special case of sparse optimization by stating the following problem [14], [74]:

$$\begin{aligned} \text{Min} \quad & C\left(\sum_{i=1}^{m_1} y_i+\sum_{l=1}^{m_2} z_l\right)+\|w\|_0 \\ \text{subject to} \quad & \\ & -a_i^T w+\gamma+1 \leq y_i, \quad i=1,\ldots,m_1 \\ & b_l^T w-\gamma+1 \leq z_l, \quad l=1,\ldots,m_2 \\ & y_i, z_l \geq 0 \end{aligned} \tag{3.11}$$

where $\|.\|_0$ is the l_0-pseudo-norm, which counts the number of nonzero components of any vector. This problem is equivalent to the following parametric program [69]:

$$\begin{aligned} \text{Min} \quad & C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) + \sum_{i=1}^{n} s\left(v_i\right) \\ & \text{subject to} \\ & -a_i^T w + \gamma + 1 \leq y_i, \quad i = 1, \ldots, m_1 \\ & b_l^T w - \gamma + 1 \leq z_l, \quad l = 1, \ldots, m_2 \\ & -v \leq w \leq v \\ & y, z \geq 0 \end{aligned} \tag{3.12}$$

Where $s : \mathbb{R} \to \mathbb{R}$ is the step function such that $s\left(t\right) = 1$ for $t > 0$ and $s\left(t\right) = 0$ for $t \leq 0$ (see also equation 3.8). This is the fundamental feature selection problem in the general setting of sparse optimization, as defined in [86].
A simplification of the models (3.11) and (3.12) can be obtained by replacing the l_0-pseudo-norm with the l_1-norm, thus obtaining:

$$\begin{aligned} \text{Min} \quad & C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) + e^T v \\ & \text{subject to} \\ & -a_i^T w + \gamma + 1 \leq y_i, \quad i = 1, \ldots, m_1 \\ & b_l^T w - \gamma + 1 \leq z_l, \quad l = 1, \ldots, m_2 \\ & -v \leq w \leq v \\ & y, z \geq 0 \end{aligned} \tag{3.13}$$

It has been demonstrated that model (3.13) exhibits in practice good sparsity properties of the solution.

3.4 Feature Selection in SVM by using the *k*-Norm

We consider, in a general setting, the following sparse optimization problem:

$$\begin{aligned} & \underset{\mathbf{x} \in \mathbb{R}^n}{\text{Minimize}} \quad f(x) + \|x\|_0 \\ & \text{subject to} \\ & \qquad x \in P, \end{aligned} \tag{3.14}$$

where we assume that $f : \mathbb{R}^n \to \mathbb{R}$ is convex and $f\left(x\right) \geq 0$ for all $x \in \mathbb{R}^n$, as it is the case when f is the error function in the SVM model. We introduce now the k-norm.

A class of polyhedral norms (the k-norms), is introduced to obtain sparse approximation solutions to systems of linear equations. In more recent years the use of k-norms has received much attention and has led to several proposals for dealing with l_0-pseudo-norm cardinality constrained problem [14].

Definition 3.3. . ***(k-norm)**. [14]. The k-norm is defined as the sum of k largest components (in modulus) of the vector X:*

$$\begin{aligned} \|x\|_{[k]} =& \mid x_{i1} \mid + \mid x_{i2} \mid + \ldots + \mid x_{ik} \mid \\ & where \mid x_{i1} \mid \geq \mid x_{i2} \mid \geq \ldots \geq \mid x_{in} \mid \end{aligned} \tag{3.15}$$

The k-norm is polyhedral, it is intermediate between $\|.\|_1$ and $\|.\|_\infty$, and the following properties hold [14]:

- $\|x\|_\infty = \|x\|_{[1]} \leq \ldots \leq \|x\|_{[k]} \leq \ldots \leq \|x\|_{[n]} = \|x\|_1$.
- $\|x\|_0 \leq k \Rightarrow \|x\|_1 - \|x\|_{[s]} = 0, \quad k \leq s \leq n$.

The k-norm enjoys the fundamental property linking $\|.\|_{[k]}$ to $\|.\|_0$, $1 \leq k \leq n$ [14]:

$$\|x\|_0 \leq k \Leftrightarrow \|x\|_1 - \|x\|_{[k]} = 0. \tag{3.16}$$

which allows replacing any constraint of the type $\|x\|_0 \leq k$ with a difference of norms, that is a DC constraint.

Definition 3.4. . ***(Subgradient)**. [87]. We say a vector $g \in \mathbb{R}^n$ is a subgradient of $f(x) \geq 0$ at $x \in domf$ if for all $z \in domf$:*

$$f(z) \geq f(x) + g^T(z - x) \tag{3.17}$$

If f is convex and differentiable, then its gradient at x is a subgradient.

But a subgradient can exist even when f is not differentiable at x, as illustrated in figure 3.1. The same example shows that there can be more than one subgradient of a function f at point x.
There are several ways to interpret a subgradient. A vector g is a subgradient of f at x if the affine function (of z) $f(x)+g^T(z-x)$ is a global underestimator of f. Geometrically, g is a subgradient of f at x if $(g, -1)$ supports $epif$ at $(x, f(x))$, as illustrated in figure 3.2.

Definition 3.5. . ***(Subdifferentiable)**. [87]. A function f is called subdifferentiable at x if there exists at least one subgradient at x. The set of subgradients of f at point x is called the subdifferential of f at x, and is denoted $\partial f(x)$. A function f is called subdifferentiable if it is subdifferentiable at all $x \in domf$.*

We recall some differential properties of the k-norm.In particular, given any $x \in \mathbb{R}^n$, and denoting by $J_{[k]}(\bar{x}) = \{j_1, \ldots, j_k\}$ the index set of k largest absolute-value components of $\bar{x}$, a subgradient $g^{[k]} \in \partial\|\bar{x}\|_{[k]}$ can be obtained as [14], [88]:

$$g_j^{[k]} = \begin{cases} 1, & if \;\; j \in J_{[k]}(\bar{x}) \;\; and \; \bar{x}_j \geq 0 \\ -1, & if \;\; j \in J_{[k]}(\bar{x}) \;\; and \; \bar{x}_j < 0 \\ 0, & otherwise \end{cases} \tag{3.18}$$

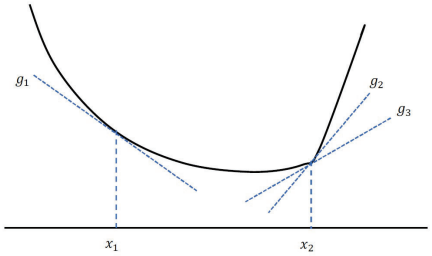

Fig. 3.1. At x_1, the convex function f is differentiable, and g_1 (which is the derivative of f at x_1) is the unique subgradient at x_1. At the point x_2, f is not differentiable. At this point, f has many subgradients: two subgradients, g_2 and g_3, are shown [87].

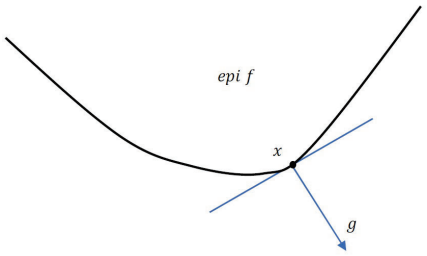

Fig. 3.2. A vector $g \in \mathbb{R}^n$ is a subgradient of f at x if and only if $(g, -1)$ defines a supporting hyperplane to $epif$ at $(x, f(x))$. [87].

Note that the subdifferential $\partial\|.\|_{[k]}$ is a singleton (i.e., the vector k-norm is differentiable) any time the set $J_{[k]}(.)$ is uniquely defined [14].
To tackle problem (3.14), in [14] and from the other observation (see [89], [90]):

$$\|x\|_{[k]} = \underset{y \in \psi_k}{\text{Maximize}} \ \ y^T x \tag{3.19}$$

where ψ_k is the subdifferential of $\partial\|.\|_{[k]}$ at point 0:

$$\psi_k = \left\{ y \in \mathbb{R}^n | y = u - v, \ 0 \leq u, v \leq e, \ (u+v)^T e = k \right\} \tag{3.20}$$

with e being the vector of n ones. Then in [14], formulated the following problem:

$$\begin{aligned} &\underset{x,u,v,z}{\text{Minimize}} \ \ f(x) + z \\ &\text{subject to} \\ &\quad e^T(u+v) = z \\ &\quad (u-v)^T x \geq \|x\|_1 \\ &\quad 0 \leq u, v \leq e, \ x \in \mathbb{R}^n, \end{aligned} \tag{3.21}$$

And then, by eliminating the scalar variable z, problem (3.21) reformulate to the following problem [14]:

$$\begin{aligned} \underset{x,u,v,z}{\text{Minimize}} \;\; & f(x) + e^T(u+v) \\ \text{subject to} \;\; & \\ & (u-v)^T x \geq \|x\|_1 \\ & 0 \leq u, v \leq e, \; x \in \mathbb{R}^n, \end{aligned} \tag{3.22}$$

By penalizing the nonlinear non-convex constraint of problem (3.22) through the scalar penalty parameter $\sigma > 0$ [14]:

$$\begin{aligned} \underset{x,u,v,z}{\text{Minimize}} \;\; & f(x) + e^T(u+v) + \sigma(\|x\|_1 - (u-v)^T x) \\ \text{subject to} \;\; & \\ & 0 \leq u, v \leq e, \; x \in \mathbb{R}^n, \end{aligned} \tag{3.23}$$

It is shown in [14] that the objective function of problem (3.22) can be converted into the DC (Difference of two Convex functions) decomposition form. In [14], the l_0-SVM problem (SVM_0) is also proposed as follows:
At first, they considered the SVM model with l_1-norm in the form of the following problem [14]:

$$\begin{aligned} \text{Min} \;\; & C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) + \|w\|_1 \\ \text{subject to} \;\; & \\ & -a_i^T w + \gamma + 1 \leq y_i, \;\; i = 1, \ldots, m_1 \\ & b_l^T w - \gamma + 1 \leq z_l, \;\; l = 1, \ldots, m_2 \\ & y_i, z_l \geq 0. \end{aligned} \tag{3.24}$$

Then by letting [14]:

$$w = w^+ - w^-, \; w^+, w^- \geq 0, \tag{3.25}$$

and indicating by e the vector of ones of dimension n, the above problem can be rewritten in a Linear Programming form as follows [14]:

$$\begin{aligned} \text{Min} \;\; & C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) + e^T(w^+ + w^-) \\ \text{subject to} \;\; & \\ & -a_i^T(w^+ - w^-) + \gamma + 1 \leq y_i, \;\; i = 1, \ldots, m_1 \\ & b_l^T(w^+ - w^-) - \gamma + 1 \leq z_l, \;\; l = 1, \ldots, m_2 \\ & y_i, z_l \geq 0, \qquad w^+, w^- \geq 0. \end{aligned} \tag{3.26}$$

Of course, problem (3.26) is an equivalent formulation of the SVM problem (3.24) [14].

In the sequel in [14] they set the sparse optimization approach through k-norm for feature selection in the SVM model and obtained the SVM_0 problem as follows:

$$\begin{aligned}
\text{Min} \quad & C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) + e^T(u+v) \\
& \text{subject to} \\
& (u-v)^T(w^+ - w^-) \geq e^T(w^+ + w^-) \\
& -a_i^T(w^+ - w^-) + \gamma + 1 \leq y_i, \quad i = 1, \ldots, m_1 \\
& b_l^T(w^+ - w^-) - \gamma + 1 \leq z_l, \quad l = 1, \ldots, m_2 \\
& y_i, z_l \geq 0, \qquad w^+, w^- \geq 0. \\
& 0 \leq u, v \leq e.
\end{aligned} \tag{3.27}$$

By penalizing the nonlinear constraint of the problem (3.27), the following problem is obtained [14]:

$$\begin{aligned}
\text{Min} \quad & C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) + e^T(u+v) + \sigma(e^T(w^+ + w^-) - (u-v)^T(w^+ - w^-)) \\
& \text{subject to} \\
& -a_i^T(w^+ - w^-) + \gamma + 1 \leq y_i, \quad i = 1, \ldots, m_1 \\
& b_l^T(w^+ - w^-) - \gamma + 1 \leq z_l, \quad l = 1, \ldots, m_2 \\
& y_i, z_l \geq 0, \qquad w^+, w^- \geq 0. \\
& 0 \leq u, v \leq e.
\end{aligned} \tag{3.28}$$

4

Multi-Objective Optimization Problems

In this chapter, we will first introduce some basic concepts in the field of multi-objective optimization problems, and then we will present some methods for solving multi-objective optimization problems. The topics of this chapter are completely taken from the [35] article.

4.1 Introduction

Optimization is a basic process in many fields of science, including mathematics, economics, management and engineering applications. In most of the real situations, decisions have to be made taking into account two or more conflicting objectives, rather than a single one. Having conflicting objectives means that it is not possible to find a feasible solution where all the objectives could reach their individual optimal but one must find the most satisfactory compromise between the objectives. These compromise solutions, in which none of the objective functions can be improved in value without impairing at least one of the others, are often referred to as Pareto optimal or Pareto efficient. The set of all objective function values at the Pareto and weak Pareto solutions is said to be the Pareto front (or efficient set) of the multi-objective optimization problem (MOP) in the objective value space [102]. In general, solving a MOP is associated with the construction of the Pareto frontier. Several MOP techniques have been developed to obtain the Pareto front. In the following we will refer to some of them.
The most widely used method for MOP is the weighted sum method. The method transforms a MOP into a single objective optimization problem by multiplying each objective function by a weighting factor and summing up all contributors. Initial work on the weighted sum method can be found in Zadeh [119]. Koski [112], [113] applied the weighted sum method to structural optimization. The ε -constraint method first appeared in [110] and is discussed in detail in Changkong and Haimes [95]. It is based on a scalarization where one of the objective functions is minimized while all the other objective functions

are bounded from above by means of additional constraints. Huang and Yang [109] extend the result of [93] to nonconvex problems. They use the hybrid scalarization.
Das [100] and Das and Dennis [98], [99] present the normal boundary intersection (NBI) method. In this method, suboptimizations are performed on normal lines to the utopia hyperplane that is defined and bounded by all anchor points. The method can also determine Pareto optimal solutions in nonconvex regions (see also [135]). Messac and Mattson [125] and Mattson and Messac [126] used physical programming for generating Pareto fronts (see also [122]). They also developed the normal constraint method [124], which generates uniformly distributed solutions along the Pareto front without missing any Pareto front regions.
As previously mentioned, the problem of finding the whole solution set of a MOP is important in applications. (see [91], [92], [94], [96], [97], [104], [105], [106], [107], [108], [111], [115], [116], [117], [118], [120], [123], [127], [128], [129], [130], [131], [133], [134], [135], [137], [138], [139], [141], [142]). Also from the large amount of relevant publications in MOP, we mention just four books, namely [97], [102], [136], [144] in which most of the theoretical and practical issues concerning MOP are comprehensively treated.

4.2 Basic concepts and notations

A multi-objective optimization problem in which at least two or more objectives are conflicting is given as follows:

$$\begin{aligned} &\text{Minimize} \quad f(x) = (f_1(x), \ldots, f_p(x)) \\ &\quad \text{subject to} \\ &\qquad \mathbf{x} \in X \end{aligned} \tag{4.1}$$

Where $X \subseteq \Re^n$, and the objective functions $f_k : \Re^n \to \Re$, $k = 1, \ldots, p$, are continuous. The image of the feasible set X under the objective function mapping f is denoted as $Y = f(X)$.
Assuming that at least two objective functions are conflicting in (4.1) then no single $x \in X$ would generally minimize every f_k simultaneously. Therefore, it is necessary to introduce a new notion of optimality or Pareto efficiency, which is useful in the multi-objective framework.

Definition 4.1. *(Dominance Vector). [102] The vector $f(x^1)$ dominates another vector $f(x^2)$ and we say x^1 dominates x^2 (denoted as $x^1 \preceq x^2$), if and only if $f_k(x^1) \leq f_k(x^2)$ for all $k = 1, \cdots, p$ and $f_i(x^1) < f_i(x^2)$ for at least one $i \in \{1, \cdots, p\}$.*

Definition 4.2. *(Pareto Optimality.) [102] A feasible solution $\widehat{x} \in X$ is called efficient or Pareto optimal to MOP* (4.1) *if there is no other $x \in X$ such that $f(x) \preceq f(\widehat{x})$. If $\widehat{x}$ is efficient, $f(\widehat{x})$ is called a nondominated point. The set of all efficient solutions $\widehat{x} \in X$ is denoted X_E and called the efficient set. The set of all nondominated points $\widehat{y} = f(\widehat{x}) \in Y$, where $\widehat{x} \in X_E$, is denoted Y_N and called the nondominated set.*

Definition 4.3. *(Weakly Pareto Optimality.) [102] A feasible solution $\widehat{x} \in X$ is called weakly efficient or weakly Pareto optimal to MOP* (4.1) *if there is no $x \in X$ such that $f(x) < f(\widehat{x})$, i.e. there is no $x \in X$ such that $f_k(x) < f_k(\widehat{x})$ for all $k = 1, \cdots, p$. If $\widehat{x}$ is weakly efficient, the point $f(\widehat{x})$ is then called weakly nondominated. The weakly efficient and nondominated sets are denoted as X_{wE} and Y_{wN}, respectively.*

Definition 4.4. *(Strictly Pareto Optimality.) [102] A feasible solution $\widehat{x} \in X$ is called strictly efficient or strictly Pareto optimal to MOP* (4.1) *if there is no $x \in X$, $x \neq \widehat{x}$ such that $f_k(x) \leq f_k(\widehat{x})$ for all $k = 1, \cdots, p$. The set of all strictly efficient points is denoted as X_{sE}.*

Definition 4.5. *(Pareto filter.) [121] For set $I \subset X$, the Pareto filter of set I is defined as $Pareto\,(I) = \{x |\ x \in I, \nexists y \in I, f\,(y) \preceq f\,(x)\}$.*

Definition 4.6. *(Ideal point.) [102] The point $y^I = (y_1^I, \ldots, y_p^I)$ in which $y_k^I = \min_{x \in X} f_k(x)$, $k = 1, \cdots, p$, (Suppose that there are finite optimal solutions for these problems) is called the ideal point of MOP* (4.1).

Definition 4.7. *(Properly efficient.) [102] A point $x \in X$ is said to be a properly efficient solution of the MOP* (4.1) *in Geoffrions's sense, if it is efficient and if there exists a scalar $M > 0$ such that, for all i, $1 \leq i \leq p$, and each $\widehat{x} \in X$ satisfying $f_i(\widehat{x}) < f_i(x)$, there exists at least one j, $1 \leq j \leq p$, such that $f_j(\widehat{x}) > f_j(x)$ and*

$$(f_i(x) - f_i(\widehat{x}))/(f_j(\widehat{x}) - f_j(x)) \leq M. \tag{4.2}$$

The set of all properly efficient solutions is denoted by X_{pE}.

4.3 Some Methods for Solving Multi-objective Optimization Problems

In this section, we review some of the methods for solving multi-objective optimization problems.

4.3.1 ε-Constraint Method

In ε-constraint method we substitute the multi-objective optimization problem (4.1) by the ε-constraint problem [35]:

$$\begin{aligned} &\text{Minimize } \ f_k(x) \\ &\quad \text{subject to} \\ &\quad\ \mathbf{x} \in X \\ &\quad\ f_j(x) \leq \varepsilon_j, \qquad j = 1, \ldots, p, j \neq k. \end{aligned} \tag{4.3}$$

where $\varepsilon \in \Re^{p-1}$.

Theorem 4.1. [102]. 1- Let x^* be an optimal solution of (4.3) for some k. Then x^* is weakly efficient.

2- Let x^* be a unique optimal solution of (4.3) for some k. Then $x^* \in X_{sE}$ (and therefore $x^* \in X_E$).

3- The feasible solution x^* is efficient if and only if there exists an $\widehat{\varepsilon} \in \Re^{p-1}$ such that x^* is an optimal solution of (4.3) for all $k = 1, \ldots, p$.

4.3.2 Modification of ε-Constraint Method

Consider a multi-objective optimization problem such as (4.1) in which $p \geq 2$. To solve this problem, we use the two-phase algorithm that described below.

Phase I:

First, we solve the following single-objective optimization problems for $k = 1, \ldots, p$:

$$\begin{aligned} &\text{Minimize } \ f_k(x) \\ &\quad \text{subject to} \\ &\quad\ \mathbf{x} \in X \end{aligned} \tag{4.4}$$

Suppose that there are finite optimal solutions for these problems. Let $x_1^*, x_2^*, \ldots, x_p^*$ be the optimal solutions of these problems, respectively. Now we define the restricted region as follows, for $k = 1, \ldots, p$:

$$\left\{ x \in X : f_k(x_k^*) \leq f_k(x) \leq (\max_{i=1,\ldots,p; i \neq k} \{f_k(x_i^*)\}) \right\}, \tag{4.5}$$

Phase II:

Step 1: (Determine the steps length). For arbitrary values $n_k \in \aleph$, we determine the steps length Δx_k as follows, for $k = 1, \ldots, p$:

$$\Delta x_k \ = \frac{(\max_{i=1,\ldots,p; i \neq k}\{f_k(x_i^*)\}) - f_k(x_k^*)}{n_k}, \tag{4.6}$$

Step 2: (Create the sets). Then, for $k = 1, \ldots, p; k \neq j$, we define the set δ_k as follows:

$$\delta_k \quad = \left\{ \delta_k^{l_k} |\ \delta_k^{l_k} = (max_{i=1,\ldots,p;i\neq k}\{f_k(x_i^*)\}) - l_k \Delta x_k;\ l_k = 0, 1, \ldots, n_k \right\}, \tag{4.7}$$

Step 3: (Solve the single-objective problems). In each stage, for any arbitrary $j \in \{1, \ldots, p\}$ we will solve the following single-objective optimization problems for $\delta_k^{l_k} \in \delta_k$ and $l_k = 0, 1, \ldots, n_k$ [35]:

$$\begin{aligned} &\text{Minimize } \ f_j(x) \\ &\quad \text{subject to} \\ &\qquad f_k(x) \leq \delta_k^{l_k}, \ \ k = 1, \ldots, p; k \neq j \\ &\qquad \mathbf{x} \in X \end{aligned} \tag{4.8}$$

Step 4: (Approximation of Pareto frontier). For a more accurate approximation of the Pareto frontier, suppose that for any $j \in \{1, \ldots, p\}$ we will solve the problem (4.8) for $\delta_k^{l_k} \in \delta_k$ and $l_k = 0, 1, \ldots, n_k$. In addition, suppose that U_j be the set of all optimal solutions obtained by solving problem (4.8) for $\delta_k^{l_k} \in \delta_k$ and $l_k = 0, 1, \ldots, n_k$. In this case, we will consider $Pareto\left(\bigcup_{j=1}^{p} U_j\right)$ as an approximation of the Pareto frontier.
Suppose that x^* is an optimal solution of (4.8), then from Theorem 4.1 it is clear that: 1- x^* is a weakly efficient solution of MOP (4.1), and 2- Let x^* be a unique optimal solution of (4.8), then x^* is a strictly efficient solution of MOP (4.1) (and therefore is an efficient solution of MOP (4.1)). Therefore this method can give the approximation of Pareto frontier.

To illustrate this approach, we consider the bi-objective optimization problem case. Let x_1^* be the optimal solution of problem:

$$\begin{aligned} &\text{Minimize } \ f_1(x) \\ &\quad \text{subject to} \\ &\qquad \mathbf{x} \in X \end{aligned} \tag{4.9}$$

and x_2^* be the optimal solution of problem:

$$\begin{aligned} &\text{Minimize } \ f_2(x) \\ &\quad \text{subject to} \\ &\qquad \mathbf{x} \in X \end{aligned} \tag{4.10}$$

As it is seen below in Figure 4.1, by considering the lines that pass through points $f_1(x_1^*)$ and $f_1(x_2^*)$ (or, the lines that pass through points $f_2(x_2^*)$ and $f_2(x_1^*)$), the feasible region of this problem will be restricted.

For an arbitrary $n \in \aleph$, steps length Δx_1 and Δx_2 are determined as follows:

$$\Delta x_k = \frac{f_1(x_2^*) - f_1(x_1^*)}{n}, \ \ \Delta x_2 = \frac{f_2(x_1^*) - f_2(x_2^*)}{n}, \tag{4.11}$$

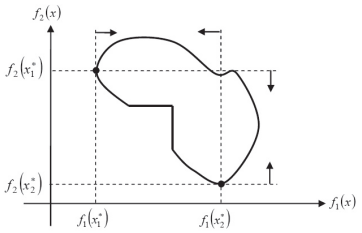

Fig. 4.1. Decision space and restricted region [35].

Then we will solve the optimization problems $(P1)$ [35]:

$$\begin{aligned} &\text{Minimize} \quad f_2(x) \\ &\text{subject to} \\ &\quad f_1(x) \leq \delta_1^{l_1} \\ &\quad \mathbf{x} \in X \end{aligned} \tag{4.12}$$

and $(P2)$:

$$\begin{aligned} &\text{Minimize} \quad f_1(x) \\ &\text{subject to} \\ &\quad f_2(x) \leq \delta_2^{l_2} \\ &\quad \mathbf{x} \in X \end{aligned} \tag{4.13}$$

for all $x \in X$ and $l_1, l_2 = 0, 1, \ldots, n$. The step lengths and the Pareto points generated (Solid circle for problem $(P1)$ and circle for problem $(P2)$), are shown in Figure 4.2 (a). The approximation of Pareto frontier is shown in Figure 4.2 (b).

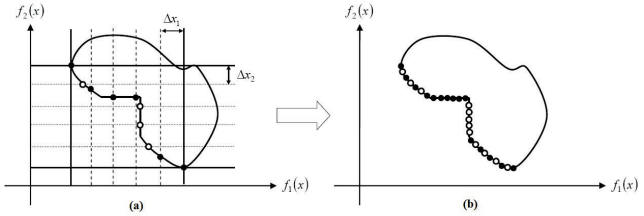

Fig. 4.2. (a) Step lengths and the Pareto points generated, (b) The approximation of Pareto frontier [35].

Theorem 4.2. [35] The feasible solution $\overline{x} \in X$ is efficient for the multi-objective optimization problem (4.1) if and only if there exists steps length Δx_k, $k = 1, \ldots, p$ such that $\overline{x}$ is an optimal solution of (4.8) for all $k = 1, \ldots, p$.

Remark 4.1. [35] We can normalize the objective functions so that all objective functions have a minimum at zero and a maximum at 1. If the objective function is unbounded or does not attain its maximum, a user-defined upper-bound can be imposed. In optimization problems with two objective functions, the normalization is done by dividing each objective function by its maximum function value when the other objective functions are at their individual minimum. Suppose that f_k^N are normalized objective functions f_k for $k = 1, \ldots, p$. In this case we determine the steps length $\Delta^{Normalize} x_k$ for $k = 1, \ldots, p$, $k \neq j$ as follows:

$$\Delta^{Normalize} x_k = \frac{1}{n_k} \tag{4.14}$$

where $n_k \in N$. Then, for $k = 1, \ldots, p; k \neq j$, we can define the set $\delta_k^{Normalize}$ as follows:

$$\delta_k^{Normalize} = \left\{ \delta_k^{l_k} |\ \delta_k^{l_k} = 1 - l_k \frac{1}{n_k};\ l_k = 0, 1, \ldots, n_k \right\}, \tag{4.15}$$

5

A New Sparse Algorithm with Application in SVM Feature Selection

This chapter introduces new Feature Selection approach based on the use of k-norm in Sparse Optimization. Then we will present new and some of the previous models in the form of multi-objective optimization problems. Some numerical experiments will be presented to compare the results of different models in single objective and multi-objective form.

5.1 A new approach to Feature Selection

This section introduces a new Feature Selection approach based on the use of k-norm in Sparse Optimization (see [14]). Then a relaxation for the model is provided. Finally, some differential properties and some algorithms for solving the proposed nonlinear model are introduced.

5.1.1 New Feature Selection by using the k-norm

We consider, in a general setting, the following sparse optimization problem:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{Minimize}} \;\; & f(x) + \|x\|_0 \\ \text{subject to} \;\; & \\ & x \in P, \end{aligned} \tag{5.1}$$

where we assume that $f : \mathbb{R}^n \to \mathbb{R}$ is convex and $f(x) \geq 0$ for all $x \in \mathbb{R}^n$, as it is the case when f is the error function in the SVM model. We introduce now the k-norm.

Definition 5.1. *. (k-norm). [14]. The k-norm is defined as the sum of k largest components (in modulus) of the vector X:*

$$\begin{aligned} \|x\|_{[k]} = & \mid x_{i1} \mid + \mid x_{i2} \mid + \ldots + \mid x_{ik} \mid \\ & where \mid x_{i1} \mid \geq \mid x_{i2} \mid \geq \ldots \geq \mid x_{in} \mid \end{aligned} \tag{5.2}$$

The k-norm is polyhedral, it is intermediate between $\|.\|_1$ and $\|.\|_\infty$ and enjoys the fundamental property linking $\|.\|_{[k]}$ to $\|.\|_0$, $1 \leq k \leq n$:

$$\|x\|_0 \leq k \Leftrightarrow \|x\|_1 - \|x\|_{[k]} = 0. \tag{5.3}$$

The property above is used to define the following Mixed Integer Nonlinear Programming (MINLP) formulation of problem (5.1), where we have introduced the set of binary variables y_k, $k = 1, \ldots, n$.

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{Minimize}} \;\; & f(x) - \sum_{k=1}^{n} y_k \\ \text{subject to} \;\; & \\ & \|x\|_{[k]} \geq \|x\|_1 y_k, \;\; k = 1, \ldots, n \\ & x \in P, \\ & y_k \in \{0, 1\}, \quad k = 1, \ldots, n. \end{aligned} \tag{5.4}$$

Note that, at the optimum of (5.4), the following hold:

$$y_k = \begin{cases} 0, & if \;\; \|x\|_{[k]} < \|x\|_1 \\ 1, & if \;\; \|x\|_{[k]} = \|x\|_1, \end{cases} \tag{5.5}$$

thus, taking into account (5.3), $y_k = 1$ if $\|x\|_0 \leq k$. Summing up we have:

$$\sum_{k=1}^{n} y_k = n - \|x\|_0 + 1 \Rightarrow \|x\|_0 = n - \sum_{k=1}^{n} y_k + 1, \tag{5.6}$$

from which we obtain that maximization of $\sum_{k=1}^{n} y_k$ implies minimization of $\|x\|_0$.

5.1.2 Relaxation of New Feature Selection Model

We can relax the integrality constraint on y_k in problem (5.4) by setting $y_k \in [0, 1]$ for $k = 1, \ldots, n$. We observe that at the optimum of the relaxed problem all constraints $\|x\|_{[k]} \geq \|x\|_1 y_k$ are satisfied by equality, which implies that for variables y_k it is $y_k = \dfrac{\|x\|_{[k]}}{\|x\|_1}$ and, consequently, they can be eliminated, obtaining:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{Minimize}} \;\; & f(x) - \frac{1}{\|x\|_1} \sum_{k=1}^{n} \|x\|_{[k]} \\ \text{subject to} \;\; & \\ & x \in P \end{aligned} \tag{5.7}$$

From now on, we will consider problem (5.7) as our main problem and call it "Our Model" or "BM-SVM". We rewrite it in following fractional programming form:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{Minimize}} \quad & \frac{f(x)\|x\|_1 - \sum_{k=1}^{n}\|x\|_{[k]}}{\|x\|_1} \\ \text{subject to} \quad & \\ & x \in P \end{aligned} \tag{5.8}$$

5.1.3 Some Differential Properties and Some Algorithms for Solving the Proposed Nonlinear Model

Problem above can be tackled via Dinkelbach's method [145], which consist in solving the scalar nonlinear equation $F(p) = 0$ where:

$$F(p) = \underset{x\in P\subset\mathbb{R}^n}{\text{Min}} \underbrace{f(x)\|x\|_1 - \sum_{k=1}^{n}\|x\|_{[k]} - p\|x\|_1}_{f_p(x)}. \tag{5.9}$$

Remark 5.1. Calculation of $F(p)$ amounts to solving an optimization problem in DC (Difference of Convex) form. Observe, in fact, that function $f(x)\|x\|_1$ is convex, being the product of two convex and non-negative functions. Thus function $f_p(x)$ can be put in DC form $f_p(x) = f_p^{(1)}(x) - f_p^{(2)}(x)$ by letting:

$$\begin{cases} f_p^{(1)}(x) = & f(x)\|x\|_1, \\ f_p^{(2)}(x) = \sum_{k=1}^{n}\|x\|_{[k]} + p\|x\|_1, \end{cases} \tag{5.10}$$

if $p \geq 0$, and:

$$\begin{cases} f_p^{(1)}(x) = f(x)\|x\|_1 - p\|x\|_1, \\ f_p^{(2)}(x) = & \sum_{k=1}^{n}\|x\|_{[k]}, \end{cases} \tag{5.11}$$

if $p < 0$.

Remark 5.2. Function $f_p(x)$ is nonsmooth. Thus the machinery provided by the literature on optimization of nonsmooth DC functions can be fruitfully adopted to tackle (5.9) (see [12], [13] and the references therein). We recall some differential properties of the k-norm. In particular, given any $x \in \mathbb{R}^n$, and denoting by $J_{[k]}(\bar{x}) = \{j_1, \ldots, j_k\}$ the index set of k largest absolute-value components of $\bar{x}$, a subgradient $g^{[k]} \in \partial\|\bar{x}\|_{[k]}$ can be obtained as [14], [88]:

$$g_j^{[k]} = \begin{cases} 1, & if \;\; j \in J_{[k]}(\bar{x}) \;\; and \; \bar{x}_j \geq 0 \\ -1, & if \;\; j \in J_{[k]}(\bar{x}) \;\; and \; \bar{x}_j < 0 \\ 0, & otherwise \end{cases} \tag{5.12}$$

Note that the subdifferential $\partial\|.\|_{[k]}$ is a singleton (i.e., the vector k-norm is differentiable) any time the set $J_{[k]}(.)$ is uniquely defined.
In the next section, we will present the previous models in the form of multi-objective optimization problems.

5.2 Reformulation of Feature Selection Problems into Multi-Objective Optimization Form

The MOP reformulations of the l_1 model (2.24) and l_2 model (2.23) in chapter 2 (section (2.2)), are as follows, respectively:

$$\begin{aligned}
&\text{Min} \quad \sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l \\
&\text{Min} \quad \|w\|_1 \\
&\quad \text{subject to} \\
&\qquad -a_i^T w + \gamma + 1 \le y_i, \;\; i = 1, \ldots, m_1 \\
&\qquad b_l^T w - \gamma + 1 \le z_l, \;\; l = 1, \ldots, m_2 \\
&\qquad\qquad y_i, z_l \ge 0
\end{aligned} \tag{5.13}$$

$$\begin{aligned}
&\text{Min} \quad \sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l \\
&\text{Min} \quad \|w\|_2^2 \\
&\quad \text{subject to} \\
&\qquad -a_i^T w + \gamma + 1 \le y_i, \;\; i = 1, \ldots, m_1 \\
&\qquad b_l^T w - \gamma + 1 \le z_l, \;\; l = 1, \ldots, m_2 \\
&\qquad\qquad y_i, z_l \ge 0
\end{aligned} \tag{5.14}$$

Our FS model, formulated according to problem (5.7) in section 5.1 is reformulated as the following MOP:

$$\begin{aligned}
&\text{Min} \quad \sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l \\
&\text{Min} \quad -\frac{1}{\|x\|_1} \sum_{k=1}^{n} \|x\|_{[k]} \\
&\quad \text{subject to} \\
&\qquad -a_i^T w + \gamma + 1 \le y_i, \;\; i = 1, \ldots, m_1 \\
&\qquad b_l^T w - \gamma + 1 \le z_l, \;\; l = 1, \ldots, m_2 \\
&\qquad\qquad y_i, z_l \ge 0
\end{aligned} \tag{5.15}$$

And SVM_0 model formulated according to problem (3.28) in Chapter 4 (section 3.4) reformulate as the following MOP:

$$\begin{aligned}
&\text{Min} \quad C\left(\sum_{i=1}^{m_1} y_i + \sum_{l=1}^{m_2} z_l\right) \\
&\text{Min} \quad e^T(u+v) + \sigma(e^T(w^+ + w^-) - (u-v)^T(w^+ - w^-)) \\
&\quad \text{subject to} \\
&\qquad -a_i^T(w^+ - w^-) + \gamma + 1 \leq y_i, \quad i = 1, \ldots, m_1 \\
&\qquad b_l^T(w^+ - w^-) - \gamma + 1 \leq z_l, \quad l = 1, \ldots, m_2 \\
&\qquad\qquad y_i, z_l \geq 0, \qquad w^+, w^- \geq 0. \\
&\qquad\qquad 0 \leq u, v \leq e.
\end{aligned} \tag{5.16}$$

To solve these multi-objective optimization problems, we can use a modified algorithm based on the ϵ-constraint method which was introduced in Chapter 4 (Section 4.3). The methods presented in [146] and [147] can be used as well.

5.3 Numerical Experiments

In this section, some numerical experiments are presented to compare the results of different models. We will take the results of single-objective problems and the results of MOP reformulations for some of the numerical experiments. To solve the test problems, we used the Global Solve solver of the global optimization package in MAPLE v.18.01. The algorithms in the Global Optimization toolbox are global search methods, which in this method systematically search the entire feasible region for a global extremum (see [148]).

Test Problem 1. (Single objective testing). The following two sets are given:

$$A = \{(1,4,1), (1.5,6,1), (3.5,5,1)\}, \; B = \{(2,6,3), (3,5,2), (6,3,1.7)\}$$

In this example, the number of samples is 6 and the number of features is 3. We have set $C = 10$. All models provide the correct separator of the sets (the error of all models is equal to zero). But l_1 and l_2 return a vector w where components are all nonzero, whereas the vector w returned by our sparse optimization method has just one nonzero component. The results of this example are depicted in Table 5.1 and Fig. 5.1.

Test Problem 2. (Single objective and multi-objective testing). In this example, the number of samples is 14, and the number of features is 3. Suppose that we have the following two sets:

Table 5.1. The results of Test problem 1 for $C = 10$.

Method	w_1	w_2	w_3	$\|w\|_1$	Error	Correctness
Our Model	0	0	-2.8571	2.8571	0	% 100.00
l_1 **Model**	-0.0764	0.1911	-2.0383	2.3058	0	% 100.00
l_2 **Model**	-0.0764	0.1911	-2.0383	2.3058	0	% 100.00

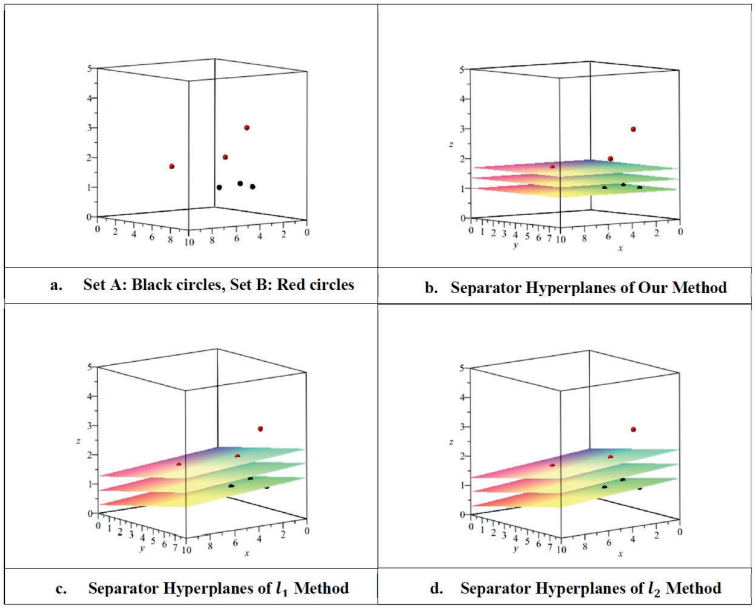

Fig. 5.1. The Result of Separator Hyperplanes for Test Problem 1, ($C = 10$).

$$A = \{[2, 5, 1], [1.7, 4, 1.5], [3, 5.5, 1.6], [2.5, 5.3, 1.3], [1.5, 1.5, 0.8], [2.5, 3.5, 1.4], [2.8, 4, 1.2]\}$$

$$B = \{[3.2, 6, 2], [3.5, 5.8, 2.4], [5, 4.1, 1.9], [4, 6.5, 3], [3.8, 8, 2], [6, 6, 2], [4.2, 6.1, 1.8]\}$$

We have set $C = 10$. All single objective models provide the correct separator of the sets (the error of all models is equal to zero). But l_1 and l_2 return a vector w where components are all nonzero, whereas the vector w returned by $BM - SVM$ (Our Model) and SVM_0 methods has just one nonzero component. The results of this example for single objective models are depicted in Table 5.2 and Figure 5.2.

Now we used this dataset for MOP models. The Algorithm introduced in Chapter 4 (section 4.3) has been used to solve these MOPs. Here we have set $d = 100$. Out of 100 Pareto solutions that were obtained for each MOP,

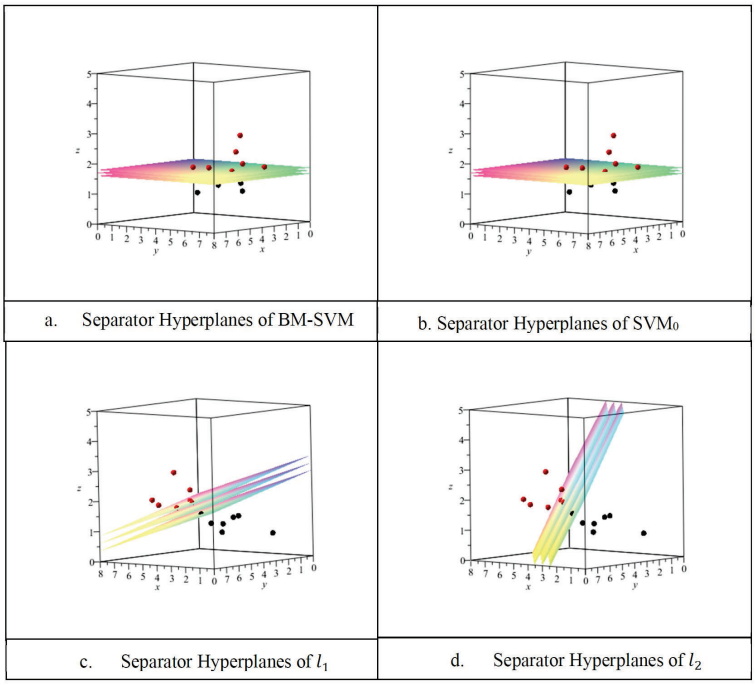

Fig. 5.2. The result of Separator Hyperplanes in single objective models for Test problem 2, ($C = 10$).

Table 5.2. The results of Test problem 2 for $C = 10$.

Method	w_1	w_2	w_3	$\|w\|_1$	Error	Correctness
Our Model	0.00	0.00	-9.9998	9.9998	0	% 100.00
SVM_0 **Model**	0.00	0.00	-10.00	10.00	0	% 100.00
l_1 **Model**	-0.7500	-0.5000	-4.0000	5.2500	0	% 100.00
l_2 **Model**	-1.8265	-1.6276	-1.9541	5.4082	0	% 100.00

we have considered 6 Pareto solutions for more consideration. In Figures 5.3-5.6, we have considered a suitable viewing angle for each specific sample (6 Pareto solutions) to have a better view of the separating hyperplanes, for MOP models. Also, in tables 5.3-5.7, the results obtained for the same Pareto optimal solutions are displayed.
In Table 5.3 for the l_1 multi-objective model, the value of $\|w\|_1$ gradually decreases in the solutions, while the error value increases.

For example, in the first and second Pareto solutions with an error value equal to zero, a smaller value for the $\|w\|_1$ has been achieved compared to the results of the single-objective l_1 problem, presented in Table 5.2. For example, in the sixth optimal solution, the value of one of the components of the vector

Table 5.3. The results of l_1 MOP model for the dataset of Test problem 2.

	w_1	w_2	w_3	$\|w\|_1$	Error	Correctness
1	-0.7400	-0.4933	-3.9470	5.1803	0.00	% 100.00
2	-0.7048	-0.4699	-3.7590	4.9337	0.00	% 100.00
3	-0.4405	-0.2937	-2.3494	3.0834	0.8253	% 92.86
4	-0.8068	-0.1502	-1.0164	1.9734	1.3569	% 85.72
5	-0.7739	-0.3108	-0.0254	1.1101	2.9710	% 64.29
6	-0.7405	-0.2462	0.00	0.9867	3.4612	% 50.00

w is equal to zero, but the error has increased.

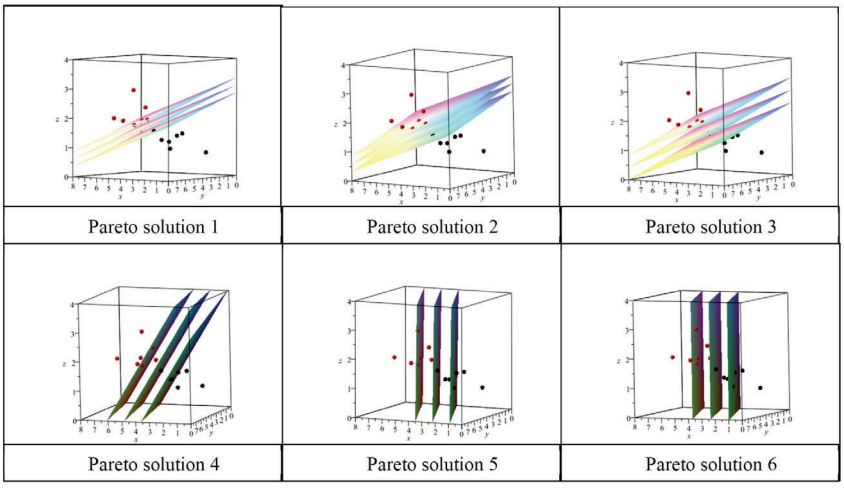

Fig. 5.3. Some results of Separator Hyperplanes in l_1 MOP model for Test problem 2.

In Table 5.4 for the l_2 multi-objective model, the value of $\|w\|_1$ gradually decreases in the solutions, while the error value increases. For example, in the first and second Pareto solutions with an error value equal to zero, a smaller value for the $\|w\|_1$ has been achieved compared to the results of the single-objective l_2 problem, presented in Table 5.2. Also, in different Pareto solutions, none of the components of the vector w become zero.

In Table 5.5 for the $BM-SVM$ multi-objective model, in the first Pareto solution with an error value equal to zero, a smaller value for the $\|w\|_1$ has been achieve but all components of the vector w are non-zero. In the second Pareto solution with an error value equal to zero, two components of the vec-

Table 5.4. The results of l_2 MOP model for the dataset of Test problem 2.

	w_1	w_2	w_3	$\|w\|_1$	Error	Correctness
1	-1.0146	-0.7756	-3.5230	5.3132	0.00	% 100.00
2	-1.7660	-1.5736	-1.8893	5.2289	0.00	% 100.00
3	-1.0196	-0.9085	-1.0908	3.0189	0.9055	% 92.86
4	-0.9476	-0.7882	-0.9616	2.6974	1.0317	% 85.72
5	-0.7108	-0.4092	-0.7411	1.8611	1.6263	% 78.57
6	-0.6313	-0.3030	-0.3473	1.2816	2.7562	% 57.14

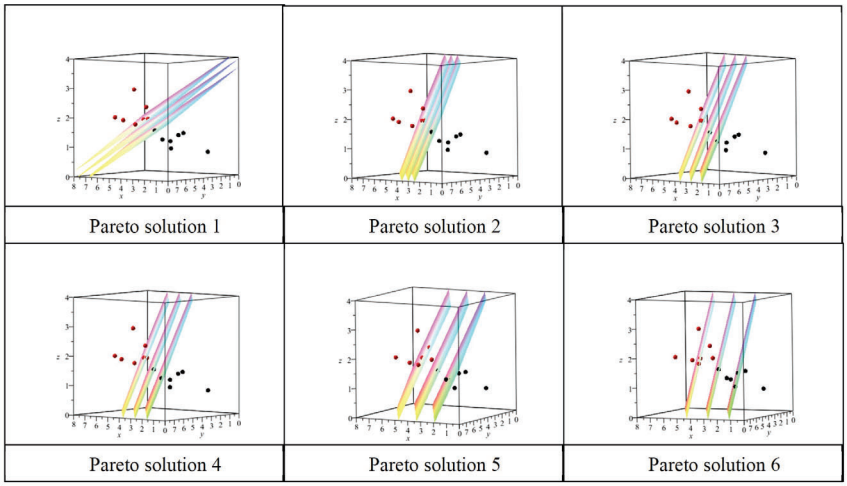

Fig. 5.4. Some results of Separator Hyperplanes in l_2 MOP model for Test problem 2.

tor w are non-zero. For the other Pareto solutions, a smaller value for the $\|w\|_1$ has been achieve and has just one nonzero component.

Table 5.5. The results of $BM-SVM$ MOP model for the dataset of Test problem 2.

	w_1	w_2	w_3	$\|w\|_1$	Error	Correctness
1	-0.7500	-0.5000	-4.0000	5.2500	0.00	% 100.00
2	-0.0460	0.00	-9.7375	9.7835	0.00	% 100.00
3	0.00	0.00	-8.5929	8.5929	0.00	% 100.00
4	-6.5340	0.00	0.00	6.5340	0.7374	% 92.86
5	0.00	0.00	-5.00	5.00	1.4748	% 85.71
6	0.00	0.00	-4.00	4.00	1.60	% 78.57

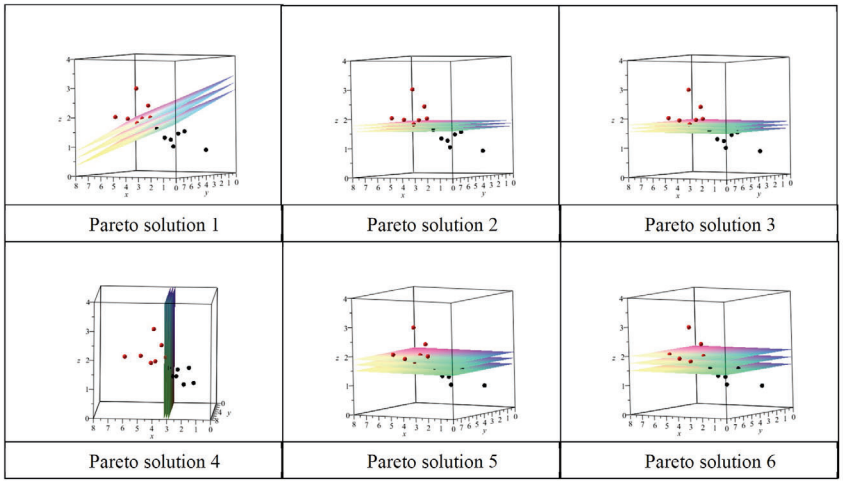

Fig. 5.5. Some results of Separator Hyperplanes in $BM - SVM$ MOP model for Test problem 2.

For SVM_0 multi-objective model, as shown in Figure 5.6 and Table 5.6, in the first Pareto solution with an error value equal to zero, a smaller value for the $\|w\|_1$ has been achieve but all components of the vector w are non-zero. In the second Pareto solution with an error value equal to zero, two components of the vector w are non-zero. For the other Pareto solutions, a smaller value for the $\|w\|_1$ has been achieve and has just one nonzero component.

Table 5.6. The results of SVM_0 MOP model for the dataset of Test problem 2.

	w_1	w_2	w_3	$\|w\|_1$	Error	Correctness
1	-0.7500	-0.5000	-4.0000	5.2500	0.00	% 100.00
2	-5.00	0.00	-2.5000	7.5000	0.00	% 100.00
3	0.00	0.00	-9.8603	9.8603	0.00	% 100.00
4	0.00	0.00	-7.2591	7.2591	0.5481	% 92.86
5	-6.5625	0.00	0.00	6.5625	1.2386	% 92.86
6	0.00	0.00	-5.00	5.00	2.1090	% 78.57

All Pareto optimal solutions (in the functions space of Error (Vertical axis) an l_1 norm (Horizontal axis)) obtained from multi-objective models ($BM - SVM$, SVM_0, l_1, l_2) are shown in Figure 5.7.

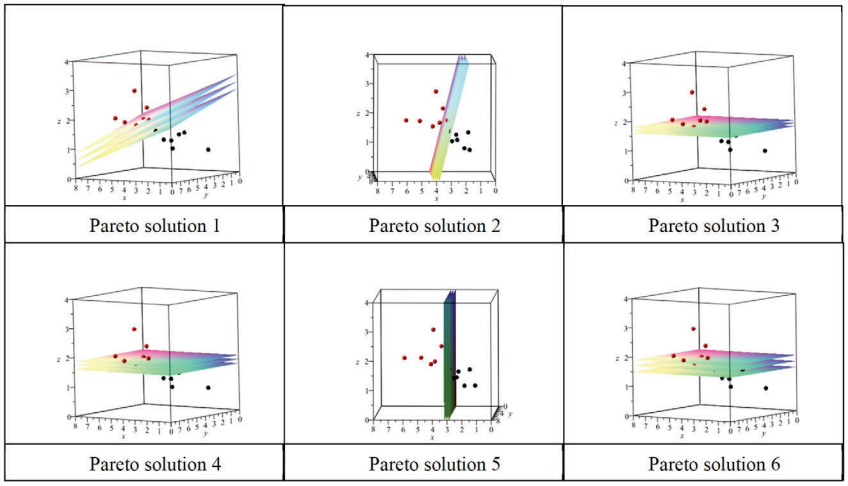

Fig. 5.6. Some results of Separator Hyperplanes in SVM_0 MOP model for Test problem 2.

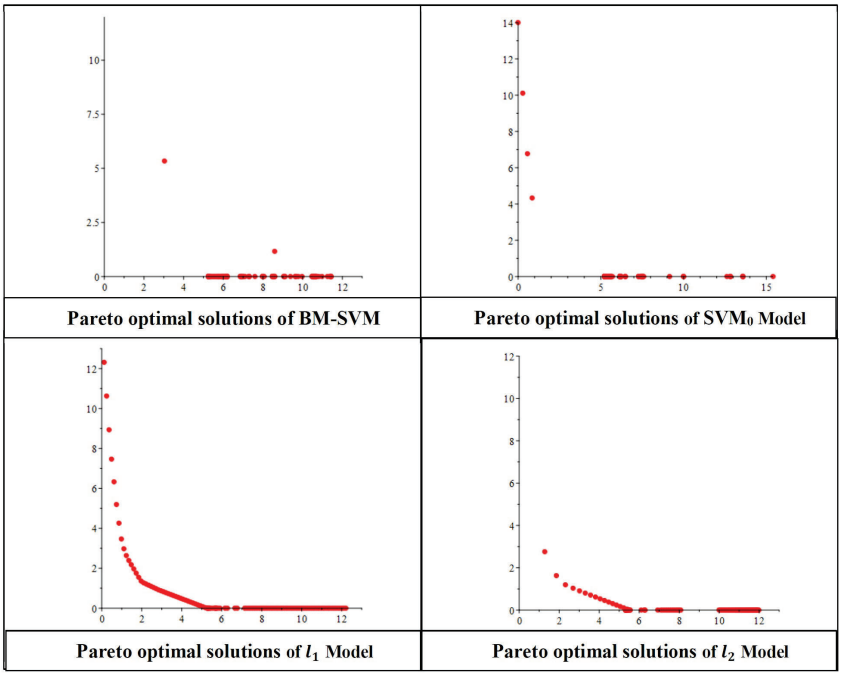

Fig. 5.7. Pareto optimal solutions obtained from multi-objective models ($BM - SVM$, SVM_0, l_1, l_2 models) for the dataset of Test problem 2.

Test Problem 3. (Single objective and multi-objective testing). In this example, the number of samples is 6, and the number of features is 4. Suppose that we have the following two sets:

$$A = \{[1.5, 4.2, 1, 2], [1.9, 4.6, 1.5, 1.5],\ [1.8, 4.5, 1.6, 1.9]\}$$

$$B = \{[2.2, 6, 3, 2.1], [2.6, 5, 2, 2.3], [4, 4.7, 1.7, 2.5]\}$$

We have set $C = 10$, and in this test problem also all models provide the correct separator of the sets. Models l_1 and l_2 return a vector w where components are all nonzero, but the vector w returned by our method has just one nonzero component. The results of this test problem is shown in Table 5.11.

Table 5.7. The results of Single Objective Models for Test problem 3 for $C = 10$.

Method	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
Our Model	-6.8067	0	0	0	6.8067	0	% 100.00
l_1 **Model**	-1.9444	-0.0001	-0.8333	-0.2778	3.0556	0	% 100.00
l_2 **Model**	-1.3223	-0.8264	-0.6612	-0.6612	3.4711	0	% 100.00

Now we used the dataset of Test problem 3 for MOP models. We have used the algorithm introduced in chapter 4 (section 4.3) to solve these MOPs, and in this algorithm, we have set $d = 100$. Out of 100 Pareto solutions that were obtained for each model we have considered only 2 Pareto solutions that seemed interesting for more consideration that are displayed in Tables 5.8, 5.9 and 5.10.

For the l_1 MOP model, as shown in Table 5.8, for the first Pareto solution with an error value equal to zero we have obtained a smaller value for $\|w\|_1$, compared to the results of the l_1 single-objective model presented in Table 5.11. For the second Pareto solution, we have obtained a solution where one of the components of w is equal to zero, but the error value is non-zero.

For the l_2 MOP model, as shown in Table 5.9, for the first Pareto solution

Table 5.8. The results of 2 Pareto solutions of l_1 MOP model for the dataset of Test problem 3.

	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
1	-1.9444	0	-0.8332	-0.2778	3.0554	0	% 100.00
2	-1.3665	0	-0.7177	-0.8557	2.9399	0.2774	% 83.33

with an error value equal to zero a smaller value for $\|w\|_1$ is obtained, com-

pared to the results of the l_2 single-objective model presented in Table 5.11. For the second Pareto solution we have obtained smallest value for $\|w\|_1$, but the error value is non-zero.

Table 5.9. The results of 2 Pareto solutions of l_2 MOP model for the dataset of Test problem 3.

	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
1	-1.7026	-0.7570	-0.0447	-0.6037	3.1080	0	% 100.00
2	-1.1240	-0.7025	-0.5619	-0.5620	2.9504	0.3000	% 83.33

For our MOP model, as shown in Table 5.10, the first Pareto solution is similar to the solution that was obtained in the single-objective model in which the vector w returned only one nonzero component. For the second Pareto solution also three components of vector w are equal to zero while $\|w\|_1$ has decreased but the error value is non-zero.
.

Table 5.10. The results of 2 Pareto solutions of Our MOP model for the dataset of Test problem 3.

	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
1	-9.2254	0	0	0	9.2254	0	% 100.00
2	-5.7389	0	0	0	5.7389	0.2800	% 83.33

Test Problem 4. (Single objective and multi-objective testing). In this example, the number of samples is 12, and the number of features is 4. Suppose that we have the following two sets:

$$A = \{[1.5, 4.2, 1, 2], [1.9, 4.6, 1.5, 1.5], [1.8, 4.5, 1.6, 1.9], \\ [1.5, 4.3, 1.2, 1.8], [1.2, 4.5, 1.6, 1.6], [1.7, 4.5, 1.4, 2]\}$$

$$B = \{[2.2, 6, 3, 2.1], [2.6, 5, 2, 2.3], [4, 4.7, 1.7, 2.5], \\ [3.2, 4.5, 2.1, 2.3], [3.5, 5.3, 2.5, 3.1], [2.1, 5.6, 2.5, 3.2]\}$$

We have set $C = 10$. All single objective models provide the correct separator of the sets (the error of all models is equal to zero). But l_1 and l_2 return a vector w where components are all nonzero, whereas the vector w returned by $BM - SVM$ and SVM_0 methods has just one nonzero component. The results of this example for single objective models are depicted in Table 5.11.

Table 5.11. The results of Single Objective Models for Test problem 4 for $C = 10$.

Method	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
Our Model	-10.00	0.00	0.00	0.00	10.00	0.00	% 100.00
SVM_0 **Model**	-10.00	0.00	0.00	0.00	10.00	0.00	% 100.00
l_1 **Model**	-1.7886	-0.4878	-0.3252	-0.4878	3.0894	0	% 100.00
l_2 **Model**	-1.3223	-0.8264	-0.6612	-0.6612	3.4711	0	% 100.00

Pareto optimal solutions obtained from four multi-objective models ($BM-SVM$, SVM_0 and l_1, l_2 models) are shown in Figure 5.8. In this figure, the horizontal axis represents the value of l_1 norm of vector w, and the vertical axis represents the error level in each Pareto optimal solutions.
Out of the 100 Pareto optimal solutions obtained for each MOP model, we have considered only 3 Pareto optimal solutions that seemed more interesting for further consideration. The results are displayed in Tables 5.12-5.15.

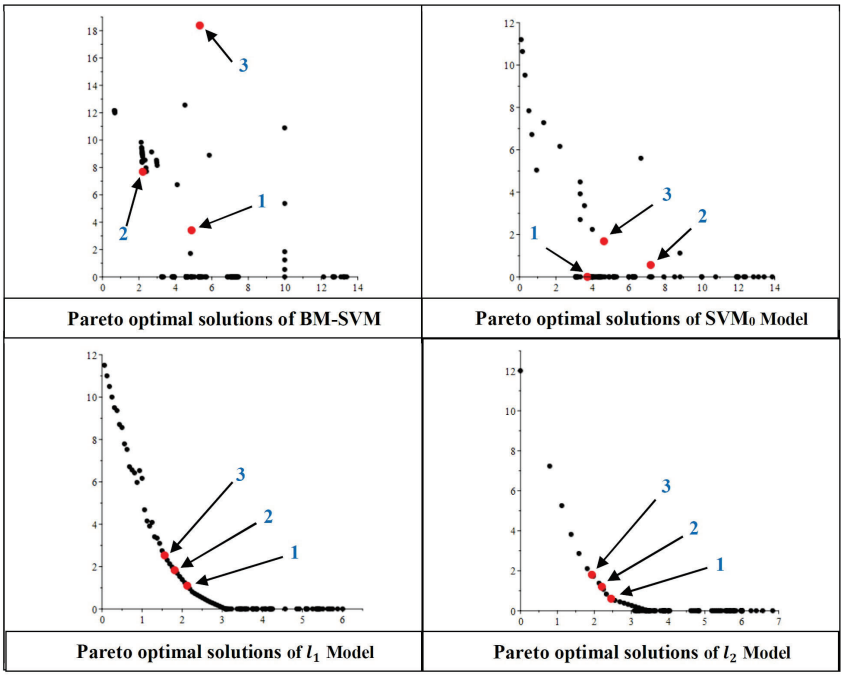

Fig. 5.8. Pareto optimal solutions obtained from multi-objective models for the dataset of Test problem 4.

For the l_1 MOP model, as shown in Table 5.12, for all pareto solutions with non-zero error value, one components of vector w equal to zero. For the l_2 MOP model, as shown in Table 5.13, for all pareto solution which are considered, all components of the vector w are non-zero.

Table 5.12. The results of some Pareto solutions of l_1 multi-objective model for the dataset of Test problem 4.

	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
1	-0.7279	0.00	-0.9160	-0.4826	2.1265	1.0984	% 66.67
2	-0.6897	0.00	-0.8314	-0.2927	1.8138	1.8273	% 58.33
3	-0.8121	0.00	-0.7514	0.00	1.5636	2.5273	% 50.00

Table 5.13. The results of some Pareto solutions of l_2 multi-objective model for the dataset of Test problem 4.

	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
1	-0.9504	0.3632	-0.5770	-0.5682	2.4588	0.60	% 83.33
2	-0.6877	0.4267	-0.6012	-0.4909	2.2065	1.20	% 66.66
3	-0.5831	0.3852	-0.5331	0.4354	1.9368	1.80	% 58.33

For $BM - SVM$ MOP model, as shown in Table 5.14, for all pareto solution which is considered, three components of vector w is equal to zero, and in each solutions smaller value for $\|w\|_1$ has been achieve but the errors are not equal to zero.

Table 5.14. The results of some Pareto solutions of $BM - SVM$ multi-objective model for the dataset of Test problem 4.

	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
1	0.00	0.00	-4.8920	0.00	4.8920	3.4020	% 75.00
2	0.00	0.00	-2.2222	0.00	2.2222	7.6788	% 66.67
3	0.00	0.00	-5.3441	0.00	5.3441	18.3710	% 58.33

For SVM_0 MOP model, as shown in Table 5.15, for the first Pareto solution which are considered, two components of vector w is equal to zero and for the odder Pareto solutions, three components of vector w is equal to zero but the error value is non-zero.

Table 5.15. The results of some Pareto solutions of SVM_0 multi-objective model for the dataset of Test problem 4.

	w_1	w_2	w_3	w_4	$\|w\|_1$	Error	Correctness
1	-1.6949	-2.0339	0.00	0.00	3.7288	0.00	% 100.00
2	-7.2008	0.00	0.00	0.00	7.2008	0.5598	% 91.67
3	0.00	0.00	-4.6410	0.00	4.6410	1.6795	% 83.33

Test Problem 5. (Single objective and multi-objective testing). In this example, the number of samples is 8, and the number of features is 5. Suppose that we have the following two sets:

$$A = \{[2.3, 3.5, 1, 2.7, 1], [2.8, 3.6, 1.5, 2.5, 1.1], \\ [2, 4.9, 1.6, 2.4, 1.2], [2.5, 3.9, 1.8, 2, 1.3]\}$$

$$B = \{[3.1, 5.6, 3, 3.1, 2], [3.6, 4.6, 2, 3.3, 2.1], \\ [4, 5, 1.7, 2.9, 2.2], [3.2, 4.2, 2.3, 2.5, 2.4]\}$$

We have set $C = 10$. All single objective models provide the correct separator of the sets (the error of all models is equal to zero). But l_1 and l_2 return a vector w where components are all nonzero, whereas the vector w returned by $BM - SVM$ and SVM_0 methods has just one nonzero component. The results of this example for single objective models are depicted in Table 5.16.

Table 5.16. The results of Single Objective Models for Test problem 5 for $C = 10$.

Method	w_1	w_2	w_3	w_4	w_5	$\|w\|_1$	Error	Correctness
Our Model	0.00	0.00	0.00	0.00	-8.3334	8.3334	0.00	% 100.00
SVM_0 **Model**	0.00	0.00	0.00	0.00	-10.00	10.00	0.00	% 100.00
l_1 **Model**	-0.1892	-0.0946	-0.2270	-0.4541	-1.3623	2.3273	0.00	% 100.00
l_2 **Model**	-0.6114	-0.2620	-0.4367	-0.4367	-0.9607	2.7074	0.00	% 100.00

Pareto optimal solutions obtained from multi-objective models ($BM - SVM$, SVM_0 and l_1, l_2 models) are shown in Figure 5.9. In this figure, the horizontal axis represents the value of l_1 norm of vector w, and the vertical axis represents the error level in each Pareto optimal solutions.
Out of the 100 Pareto optimal solutions obtained for each MOP model, we have considered only 3 Pareto optimal solutions that seemed more interesting for further consideration. The results are shows in Tables 5.17-5.20.

For the l_1 MOP model, as shown in Table 5.17, for first Pareto solution with a non-zero of error, one components of vector w equal to zero. For second

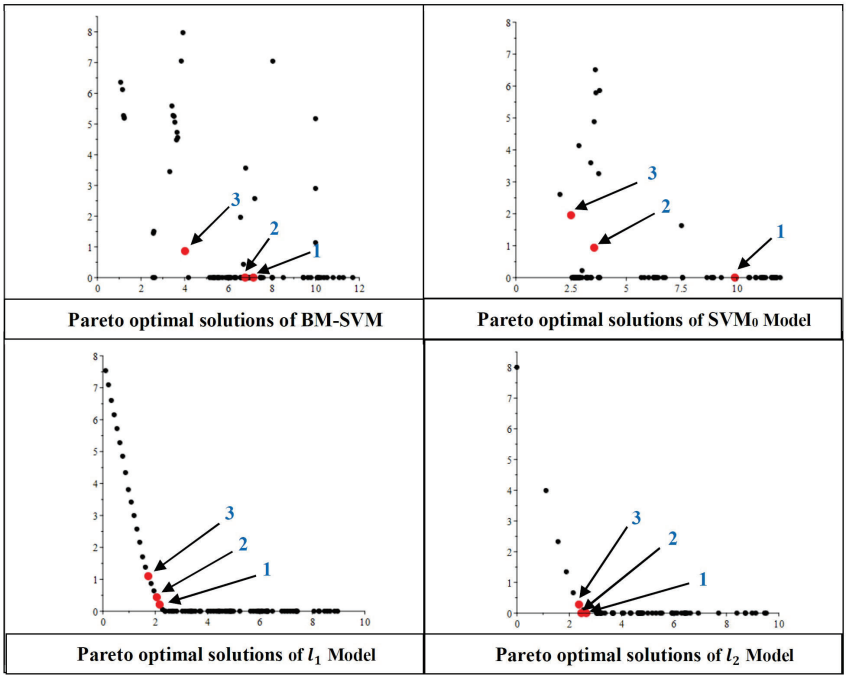

Fig. 5.9. Pareto optimal solutions obtained from multi-objective models for the dataset of Test problem 5.

Pareto solution we have obtained the smallest value for $\|w\|_1$, where one of the components of w equal to zero, and the error value is non-zero and for third Pareto solution with one component equal to zero in the vector w, the value of l_1 norm of vector w has decreased.

Table 5.17. The results of some Pareto solutions of l_1 multi-objective model for the dataset of Test problem 5.

	w_1	w_2	w_3	w_4	w_5	$\|w\|_1$	Error	Correctness
1	-0.6347	-0.2952	-0.0858	0.00	-1.1541	2.1699	0.2064	% 75.00
2	-0.6302	-0.2470	0.00	-0.0039	-1.1801	2.0612	0.4369	% 62.50
3	-0.3035	0.0437	0.00	0.00	-1.3887	1.7359	1.0963	% 50.00

For the l_2 MOP model, as shown in Table 5.18, for all Pareto solution none of the components of the vector w is equal to zero.

Table 5.18. The results of some Pareto solutions of l_2 multi-objective model for the dataset of Test problem 5.

	w_1	w_2	w_3	w_4	w_5	$\|w\|_1$	Error	Correctness
1	-0.2549	-0.1032	-0.2135	-0.7962	-1.2810	2.6489	0.00	% 100.00
2	-0.2120	-0.0556	0.3538	-0.6104	-1.2298	2.4616	0.00	% 100.00
3	-0.5363	-0.2690	-0.3751	-0.3867	-0.8051	2.3721	0.2774	% 87.50

For $BM - SVM$ MOP model, as shown in Table 5.19, for the first and second Pareto solutions which is considered, four components of vector w is equal to zero while the error values are zero. For the third Pareto solution which is considered from our MOP model, four components of vector w is equal to zero while the correctness of the model has been reduced to % 87.5.

Table 5.19. The results of some Pareto solutions of $BM - SVM$ multi-objective model for the dataset of Test problem 5.

	w_1	w_2	w_3	w_4	w_5	$\|w\|_1$	Error	Correctness
1	0.00	0.00	0.00	0.00	-7.1575	7.1575	0.00	% 100.00
2	-6.7742	0.00	0.00	0.00	0.00	6.7742	0.00	% 100.00
3	0.00	0.00	0.00	0.00	-4.0209	4.0209	0.8613	% 87.50

For SVM_0 MOP model, as shown in Table 5.15, for the first Pareto solution which is considered, four components of vector w is equal to zero and error value is equal to zero, for the second and third Pareto solutions, four components of vector w is equal to zero while the error value is non-zero.

Table 5.20. The results of some Pareto solutions of SVM_0 multi-objective model for the dataset of Test problem 5.

	w_1	w_2	w_3	w_4	w_5	$\|w\|_1$	Error	Correctness
1	0.00	0.00	0.00	0.00	-9.9147	9.9147	0.00	% 100.00
2	-3.5484	0.00	0.00	0.00	0.00	3.5484	0.9355	% 87.50
3	0.00	0.00	0.00	0.00	-2.50	2.50	1.9528	% 75.00

Test Problem 6. (Single objective testing for Benchmark Problems). We have performed our experiments on a group of five datasets adopted as benchmarks for the feature selection method described in [12]. These datasets are available at (https://www.csie.ntu.edu.tw/ cjlin/libsvmtool

s/datasets/). They are listed in Table 5.21, where m is the number of samples and n is the number of features.

A standard tenfold cross-validation has been performed in datasets. The

Table 5.21. Description of the datasets.

Datasets Names	**m**	**n**
Breast Cancer (BC)	683	10
HEART	270	13
Ionosphere (IONO)	351	34
Liver disorders (LIVER)	345	5
Sonar, Mines vs. Rocks (SONAR)	208	60

results are in Tables 5.22, 5.23 and 5.24, where the Average Training Correctness (ATC) column is expressed as the average percentage of samples correctly classified.

Columns $\|w\|_1$ report the average l_1 norm of w. Finally, columns "$\%ft(0)$"–"$\%ft(-8)$" report the average percentage of components of w whose modulus is greater than or equal to 10^0–10^{-8}, respectively. Note that, assuming, conventionally, to be equal to "zero", any component w_j of w such that $w_j < 10^{-8}$, the percentage of zero-components is $(100 - \%ft(-8))$. Two different values of parameter C have been adopted for all datasets, $C = 1$ and 10.

Table 5.22. The results of l_1 model for two values of C.

Dataset	$\|w\|_1$	ATC	Error	$\%ft(0)$	$\%ft(-2)$	$\%ft(-4)$	$\%ft(-6)$	$\%ft(-8)$
$C = 1$								
BC	1.34	%94.34	38.82	0.00	86.00	90.00	90.00	100.00
HEART	3.77	%68.27	80.31	0.77	84.62	99.23	100.00	100.00
IONO	18.11	%80.63	65.08	16.76	84.12	92.06	95.59	97.06
LIVER	0.22	%54.68	69.81	0.00	80.00	100.00	100.00	100.00
SONAR	21.57	%56.42	87.20	13.33	61.67	63.33	91.66	96.67
$C = 10$								
BC	1.38	%94.43	38.76	0.00	88.00	89.00	89.00	99.00
HEART	4.10	%69.92	79.39	1.54	86.15	100.00	100.00	100.00
IONO	35.64	%84.33	56.46	44.71	94.71	96.18	96.47	97.06
LIVER	0.23	%62.58	60.46	0.00	80.00	100.00	100.00	100.00
SONAR	152.38	%88.33	22.97	54.67	83.50	90.83	95.83	98.33

As shown in column $\%ft(-8)$ of Tables 5.22, 5.23 and 5.24, our model resets the number of more components of the w vector equal to zero in all

Table 5.23. The results of l_2 model for two values of C.

Dataset	$\|w\|_1$	ATC	Error	$\%ft(0)$	$\%ft(-2)$	$\%ft(-4)$	$\%ft(-6)$	$\%ft(-8)$
$C = 1$								
BC	1.34	%94.34	38.93	0.00	86.00	90.00	90.00	100.00
HEART	3.73	%68.72	78.63	0.00	85.38	100.00	100.00	100.00
IONO	15.83	%75.42	73.09	8.53	96.47	97.06	97.06	97.06
LIVER	0.23	%53.68	79.81	0.00	80.00	100.00	100.00	100.00
SONAR	24.99	%45.33	73.01	10.67	95.00	100.00	100.00	100.00
$C = 10$								
BC	1.39	%94.43	38.82	0.00	89.00	90.00	90.00	100.00
HEART	4.11	%69.75	77.03	1.54	86.15	100.00	100.00	100.00
IONO	32.51	%84.87	56.78	37.65	96.76	97.06	97.06	97.06
LIVER	0.23	%61.27	60.86	0.00	80.00	100.00	100.00	100.00
SONAR	62.90	%71.85	41.04	48.33	100.00	100.00	100.00	100.00

Table 5.24. The results of our model for two values of C.

Dataset	$\|w\|_1$	ATC	Error	$\%ft(0)$	$\%ft(-2)$	$\%ft(-4)$	$\%ft(-6)$	$\%ft(-8)$
$C = 1$								
BC	1.18	%96.16	34.76	0.00	64.00	64.00	64.00	64.00
HEART	3.07	%70.77	74.50	4.69	58.85	58.85	58.85	58.85
IONO	21.39	%88.23	51.33	15.29	55.46	56.80	56.80	56.80
LIVER	0.15	%67.54	64.48	0.00	60.00	70.00	70.00	70.00
SONAR	28.47	%93.88	11.27	5.00	45.00	46.67	48.33	48.33
$C = 10$								
BC	1.25	%96.46	32.17	0.00	64.00	64.00	64.00	64.00
HEART	3.12	%72.59	73.31	4.78	58.85	58.85	58.85	58.85
IONO	23.16	%89.27	50.99	15.48	55.46	56.82	56.82	56.82
LIVER	0.14	%68.52	59.45	0.00	60.00	70.00	70.00	70.00
SONAR	29.53	%95.24	10.86	5.27	45.55	46.67	48.42	48.42

datasets. Also, our model results in a smaller error value compared to the l_1 and l_2 methods. In comparison, the correctness of our model classification is better for the whole datasets. For $c = 10$, the value of $\|w\|_1$ in our model results in a smaller value in all datasets.

In the next two test problem, our goal is to demonstrate the benefits of considering the model as a multi-objective optimization problem. In this case, we can obtain a set of Pareto optimal solutions instead of one optimal solution. We consider the l_1, l_2 and our model as two-objective optimization problems described in Section 5.2.

5.4 Conclusion

In this chapter, we emphasized the application of sparse optimization in Feature Selection for Support Vector Machine classification. We have proposed a new model for sparse optimization based on the polyhedral k-norm. Due to the advantages of using multi-objective optimization models instead of single-objective models, some multi-objective reformulation of Support Vector Machine classification was proposed. The results of some test problems on classification datasets are reported for both single-objective and multi-objective models.

References

1. Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., and Vapnik, V. (2000). Feature selection for SVMs. Advances in neural information processing systems, 13.
2. Forman, G. et al. (2003). An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res., 3(Mar):1289–1305.
3. Nolfi, S., Parisi, D., and Elman, J. L. (1994). Learning and evolution in neural networks. Adaptive Behavior, 3(1):5–28.
4. Gambella, C., Ghaddar, B., and Naoum-Sawaya, J. (2021). Optimization problems for machine learning: A survey. European Journal of Operational Research, 290(3):807–828.
5. Cristianini, N., Shawe-Taylor, J., et al. (2000). An introduction to support vector machines and other kernel-based learning methods. Cambridge university press.
6. Bach, F., Jenatton, R., Mairal, J., Obozinski, G., et al. (2011). Convex optimization with sparsity-inducing norms. Optimization for Machine Learning, 5:19–53.
7. Bauschke, H. and Combettes, P. (2011). Convex analysis and monotone operator theory in hilbert spaces. CMS books in mathematics). DOI, 10:978–1.
8. Bertsimas, D., King, A., and Mazumder, R. (2016). Best subset selection via a modern optimization lens. The annals of statistics, 44(2):813–852.
9. Swain, P. H. and Davis, S. M. (1981). Remote sensing: The quantitative approach. IEEE Transactions on Pattern Analysis and Machine Intelligence, 3(06):713–714.
10. Al-Ani, A., Alsukker, A., and Khushaba, R. N. (2013). Feature subset selection using differential evolution and a wheel based search strategy. Swarm and Evolutionary Computation, 9:15–26.
11. Cervante, L., Xue, B., Zhang, M., and Shang, L. (2012). Binary particle swarm optimisation for feature selection: A filter based approach. In 2012 IEEE Congress on Evolutionary Computation, pages 1–8. IEEE.
12. Gaudioso, M., Giallombardo, G., and Miglionico, G. (2018). Minimizing piecewise-concave functions over polyhedra. Mathematics of Operations Research, 43(2):580–597.

13. Gaudioso, M., Giallombardo, G., Miglionico, G., and Bagirov, A. M. (2018b). Minimizing nonsmooth dc functions via successive dc piecewise-affine approximations. Journal of Global Optimization, 71(1):37–55.
14. Gaudioso, M., Gorgone, E., and Hiriart-Urruty, J.-B. (2020). Feature selection in SVM via polyhedral k-norm. Optimization letters, 14(1):19–36.
15. Chen, Y., Miao, D., and Wang, R. (2010). A rough set approach to feature selection based on ant colony optimization. Pattern Recognition Letters, 31(3):226–233.
16. Pilanci, M., Wainwright, M. J., and El Ghaoui, L. (2015). Sparse learning via boolean relaxations. Mathematical Programming, 151(1):63–87.
17. Wright, S. J. (2012). Accelerated block-coordinate relaxation for regularized optimization. SIAM Journal on Optimization, 22(1):159–186.
18. Watson, G. A. (1992). Linear best approximation using a class of polyhedral norms. Numerical Algorithms, 2(3):321–335.
19. Jafari-Petroudi, S. and Pirouz, M. (2016). On the bounds for the spectral norm of particular matrices with Fibonacci and Lucas numbers. Int. J. Adv. Appl. Math. and Mech, 3(4):82–90.
20. Petroudi, S. H. J., Pirouz, M., Akbiyik, M., and Yilmaz, F. (2022). Some special matrices with harmonic numbers. Konuralp Journal of Mathematics, 10(1):188–196.
21. Jafari-Petroudi, S. H. and Pirouz, B. (2015b). A particular matrix, its inversion and some norms. Appl. Comput. Math, 4:47–52.
22. Jafari-Petroudi, S. H. and Pirouz, B. (2015a). An investigation on some properties of special Hankel matrices. In The 46 the Annual Iranian Mathematics Conference, page 470.
23. Petroudi, S. H. J. and Pirouz, B. (2015). On the bounds and norms of a particular Hadamard exponential matrix. Applied mathematics in Engineering, Management and Technology, 3(2):257–263.
24. Gasso, G., Rakotomamonjy, A., and Canu, S. (2009). Recovering sparse signals with a certain family of nonconvex penalties and dc programming. IEEE Transactions on Signal Processing, 57(12):4686–4698.
25. Gotoh, J.-y., Takeda, A., and Tono, K. (2018). Dc formulations and algorithms for sparse optimization problems. Mathematical Programming, 169(1):141–176.
26. Hempel, A. B. and Goulart, P. J. (2014). A novel method for modelling cardinality and rank constraints. In 53rd IEEE Conference on Decision and Control, pages 4322–4327. IEEE.
27. Soubies, E., Blanc-F´eraud, L., and Aubert, G. (2017). A unified view of exact continuous penalties for l-2-l-0 minimization. SIAM Journal on Optimization, 27(3):2034–2060.
28. Wu, B., Ding, C., Sun, D., and Toh, K.-C. (2014). On the Moreau–Yosida regularization of the vector k-norm related functions. SIAM Journal on Optimization, 24(2):766–794.
29. Hamdani, T. M., Won, J.-M., Alimi, A. M., and Karray, F. (2007). Multi-objective feature selection with NSGA II. In International conference on adaptive and natural computing algorithms, pages 240–247. Springer.
30. Ehrgott, M. (2005). Multicriteria optimization, volume 491. Springer Science & Business Media.
31. Neshatian, K. and Zhang, M. (2009). Pareto front feature selection: using genetic programming to explore feature space. In Proceedings of the 11th Annual conference on Genetic and evolutionary computation, pages 1027–1034.

32. Dolatnezhadsomarin, A., Khorram, E., and Pourkarimi, L. (2019). Efficient algorithms for solving nonlinear fractional programming problems. Filomat, 33(7):2149–2179.
33. Ceyhan, G., K¨oksalan, M., and Lokman, B. (2019). Finding a representative nondominated set for multi-objective mixed integer programs. European Journal of Operational Research, 272(1):61–77.
34. Ghane-Kanafi, A. and Khorram, E. (2015). A new scalarization method for finding the efficient frontier in non-convex multiobjective problems. Applied Mathematical Modelling, 39(23-24):7483–7498.
35. Pirouz, B. and Khorram, E. (2016). A computational approach based on the ϵ-constraint method in multi-objective optimization problems. Adv. Appl. Stat, 49:453.
36. Pirouz, B. and Ramezani Paschapari, J. (2019). A computational algorithm based on normalization for constructing the Pareto front of multiobjective optimization problems. In 2019, the 5th International Conference on Industrial and Systems Engineering.
37. Das, I. and Dennis, J. E. (1998). Normal boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. SIAM journal on optimization, 8(3):631–657.
38. Dutta, J. and Kaya, C. Y. (2011). A new scalarization and numerical method for constructing the weak Pareto front of multi-objective optimization problems. Optimization, 60(8-9):1091–1104.
39. Fonseca, C. M., Fleming, P. J., et al. (1993). Genetic algorithms for multiobjective optimization: Formulationdiscussion and generalization. In Icga, volume 93, pages 416–423. Citeseer.
40. Pirouz, B., Ferrante, A. P., Pirouz, B., and Piro, P. (2021). Machine learning and geo-based multi-criteria decision support systems in analysis of complex problems. ISPRS International Journal of Geo-Information, 10(6):424.
41. Kartik Menon, (2022). Everything You Need to Know About Feature Selection In Machine. https://www.simplilearn.com/tutorials/machine-learning-tutorial/feature-selection-in-machine-learning
42. Feature Selection Techniques in Machine Learning. (2022). https://www.javatpoint.com/feature-selection-techniques-in-machine-learning
43. Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. ACM computing surveys (CSUR), 50(6), 1-45.
44. Tang, J., Alelyani, S., & Liu, H. (2014). Feature selection for classification: A review. Data classification: Algorithms and applications, 37.
45. Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), 1157-1182.
46. Jain, A., & Zongker, D. (1997). Feature selection: Evaluation, application, and small sample performance. IEEE transactions on pattern analysis and machine intelligence, 19(2), 153-158.
47. Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. bioinformatics, 23(19), 2507-2517.
48. Dash, M., & Liu, H. (1997). Feature selection for classification. Intelligent data analysis, 1(1-4), 131-156.
49. Liu, H., & Motoda, H. (2012). Feature selection for knowledge discovery and data mining (Vol. 454). Springer Science & Business Media.

50. Liu, H., & Motoda, H. (Eds.). (2007). Computational methods of feature selection. CRC Press.
51. Dy, J. G., & Brodley, C. E. (2004). Feature selection for unsupervised learning. Journal of machine learning research, 5(Aug), 845-889.
52. Mitra, P., Murthy, C. A., & Pal, S. K. (2002). Unsupervised feature selection using feature similarity. IEEE transactions on pattern analysis and machine intelligence, 24(3), 301-312.
53. Liu, H., Motoda, H., Setiono, R., & Zhao, Z. (2010, May). Feature selection: An ever evolving frontier in data mining. In Feature selection in data mining (pp. 4-13). PMLR.
54. Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. Artificial intelligence, 97(1-2), 273-324.
55. Ron, K., & George, H. J. (1997). Wrappers for feature subset selection. Artificial intelligence, 97(1-2), 273-324.
56. Shahana, A. H., & Preeja, V. (2016, March). Survey on feature subset selection for high dimensional data. In 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT) (pp. 1-4). IEEE.
57. Liu, H., & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. IEEE Transactions on knowledge and data engineering, 17(4), 491-502.
58. Koller, D., & Sahami, M. (1996). Toward optimal feature selection. Stanford InfoLab.
59. Liu, H., & Motoda, H. (Eds.). (2007). Computational methods of feature selection. CRC Press.
60. Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152).
61. Vapnik, V., Golowich, S., & Smola, A. (1996). Support vector method for function approximation, regression estimation and signal processing. Advances in neural information processing systems, 9.
62. Fletcher, T. (2009). Support vector machines explained. Tutorial paper, 1-19.
63. Nefedov, A. (2016). Support vector machines: a simple tutorial. Online], https://sustech-cs-courses. github. io/IDA/materials/Classification/SVM-tutorial. pdf, tanggal akses.
64. Ben-Hur, A., & Weston, J. (2010). A user's guide to support vector machines. In Data mining techniques for the life sciences (pp. 223-239). Humana Press.
65. Jakkula, V. (2006). Tutorial on support vector machine (svm). School of EECS, Washington State University, 37(2.5), 3.
66. Piccialli, V., & Sciandrone, M. (2022). Nonlinear optimization and support vector machines. Annals of Operations Research, 1-33.
67. Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines and other kernel-based learning methods. Cambridge university press.
68. Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery, 2(2), 121-167.
69. Rinaldi, F. (2009). Mathematical programming methods for minimizing the zero-norm over polyhedral sets. Sapienza, University of Rome. http://www. math. unipd. it/rinaldi/papers/thesis0. pdf.
70. Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. nature, 323(6088):533–536.

71. Haykin, S. and Network, N. (2004). A comprehensive foundation. Neural networks, 2(2004):41.
72. John, G. H., Kohavi, R., and Pfleger, K. (1994). Irrelevant features and the subset selection problem. In Machine learning proceedings 1994, pages 121–129. Elsevier.
73. Bradley, P. S. and Mangasarian, O. L. (1998). Feature selection via concave minimization and support vector machines. In ICML, volume 98, pages 82–90. Citeseer.
74. Rinaldi, F., Schoen, F., and Sciandrone, M. (2010). Concave programming for minimizing the zero-norm over polyhedral sets. Computational Optimization and Applications, 46(3):467–486.
75. Jain, V. (2010). Zero-norm optimization: Models and applications (Doctoral dissertation).
76. Bach, F., Jenatton, R., Mairal, J., & Obozinski, G. (2012). Optimization with sparsity-inducing penalties. Foundations and Trends® in Machine Learning, 4(1), 1-106.
77. Koh, K., Kim, S. J., & Boyd, S. (2007). An interior-point method for large-scale l1-regularized logistic regression. Journal of Machine learning research, 8(Jul), 1519-1555.
78. Shevade, S. K., & Keerthi, S. S. (2003). A simple and efficient algorithm for gene selection using sparse logistic regression. Bioinformatics, 19(17), 2246-2253.
79. Amaldi, E., & Kann, V. (1998). On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. Theoretical Computer Science, 209(1-2), 237-260.
80. Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), 1157-1182.
81. Warmack, R. E., & Gonzalez, R. C. (1973). An algorithm for the optimal solution of linear inequalities and its application to pattern recognition. IEEE Transactions on Computers, 100(12), 1065-1075.
82. Nakashizuka, M. (2007, July). A sparse decomposition for periodic signal mixtures. In 2007 15th International Conference on Digital Signal Processing (pp. 627-630). IEEE.
83. Mangasarian, O. L. (1996). Machine learning via polyhedral concave minimization. In Applied Mathematics and Parallel Computing (pp. 175-188). Physica-Verlag HD.
84. Frank, M., & Wolfe, P. (1956). An algorithm for quadratic programming. Naval research logistics quarterly, 3(1-2), 95-110.
85. Weston, J., Elisseeff, A., Schölkopf, B., & Tipping, M. (2003). Use of the zero norm with linear models and kernel methods. The Journal of Machine Learning Research, 3, 1439-1461.
86. Mangasarian, O. (1996). Machine learning via polyhedral concave minimization. In Applied Mathematics and Parallel Computing, pages 175–188. Springer.
87. Boyd, S., Xiao, L., & Mutapcic, A. (2003). Subgradient methods. lecture notes of EE392o, Stanford University, Autumn Quarter, 2004, 2004-2005.
88. Gaudioso, M., Gorgone, E., Labb´e, M., and Rod ´ıguez-Ch´ıa, A. M. (2017). Lagrangian relaxation for svm feature selection. Computers & Operations Research, 87:137–145.
89. Overton, M. L., & Womersley, R. S. (1993). Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices. Mathematical Programming, 62(1), 321-357.

90. Wu, B., Ding, C., Sun, D., & Toh, K. C. (2014). On the Moreau-Yosida Regularization of the Vector k-Norm Related Functions. SIAM Journal on Optimization, 24(2), 766-794.
91. Armand, P. (1993), Finding all maximal efficient faces in multi-objective linear programming. , *Math. Program. Ser.* A 61, 357–375
92. Benson, H.P. (1998), An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem, *J. Glob. Optim.* 13, 1–24
93. Benson, H.P., Morin, T.L. (1977), The vector maximization problem proper efficiency and stability, *SIAM J. Appl. Math.* 32, 64–72.
94. Benson, H.P., SUN, E. (2000), Outcome space partition of the weight set in multi-objective linear programming., *J. Optim. Theory Appl.* 105, 17–36.
95. Chankong, V., Haimes, Y. (1983), Multi-objective Decision Making Theory and Methodology., *Elsevier, New York.*
96. Cheng F, Li, D. (1999), Quality utility—a compromise programming approach to robust design, *J Mech Des.* 105, 17–36.
97. Chinchuluun, A., Pardalos, P.M, Migdalas, A., and L. Pitsoulis, (2008), Pareto Optimality, Game Theory and Equilibria, *, Springer, New York, NY.*
98. Das, I, Dennis, J.E. (1998), Normal-boundary intersection: a new method for generating Pareto optimal points in multicriteria optimization problems., *SIAM J Optim.* 8:631–657.
99. Das, I, Dennis, J.E. (1998), Normal-boundary intersection: a new method for generating the Pareto surface in nonlinear multicrieteria optimization problems., *SIAM J Optim.* 8:631–657.
100. Das, I. (1999), An improved technique for choosing parameters for Pareto surface generation using normal-boundary intersection. *In: ISSMO/UBCAD/AIASA, Third World Congress of Structural and Multidisciplinary Optimization (held in Buffalo). Buffalo: University of Buffalo, Center for Advanced Design.*
101. Dutta J., Kaya C.(2011), A new scalarization and numerical method for constructing the weak Pareto front of multi-objective optimization problem, *Optimization, Vol.60, Nos.8-9, August-September.* 1091-1104.
102. Ehrgott, M. (2005), Multicriteria Optimization, *Springer, Berlin.*
103. Ehrgott, M and S. Ruzika, (2008), Improved ε-Constraint Method for Multiobjective Programming, *J. Optim. Theory Appl,*138:375-396.
104. A. Farhang-Mehr and S. Azarm. (2002), Diversity Assessment of Pareto Optimal Solution Sets: An Entropy Approach. *In Congress on Evolutionary Computation (CEC'2002), volume 1, pages 723-728, Piscataway, New Jersey, May 2002. IEEE Service Center.*
105. Fonseca C, Fleming P (1993), Genetic algorithms for multi-objective optimization: formulation, discussion and generalization, *In: Fifth Int Conf on Genetic Algorithms,* pp 416–423.
106. Ghane, A., Khorram, E. (2015), A new scalarization method for finding the efficient frontier in non-convex multi-objective problems, *Applied Mathematical Modeling.*
107. Goldberg DE (1989), Genetic algorithms in search, optimization and machine learning,*Addison Wesley, Reading.*
108. Hernandez-Diaz, A. G.Santana-Quintero, Coello Coello, and J. Molina. (2007), Pareto-adaptive ε-dominance. Evolutionary Computation, 15(4): 493-517.

109. Huang, X.X., Yang, X.Q (2002), On characterizations of proper efficiency for nonconvex multi-objective optimization,*J. Glob. Optim. 23(3–4), 213–231* .
110. Haimes, Y.Y., Lasdon, L.S.,Wismer, D.A.,(1971), On a bicriterion formulation of the problems of integrated system identification and system optimization, *IEEE Trans. Syst. Man. Cybern,* 1, 29-297.
111. Jahn, J.(2004), Vector Optimization: Theory, Applications, and Extensions. , *Springer, Berlin.*
112. Koski J (1988), Multicriteria truss optimization,*In: Stadler W (ed) Multicriteria optimization in engineering and in the sciences. Plenum, New York.*
113. Koski J. (1985), Defectiveness of weighting methods in multicriterion optimization of structures,*Commun Appl Numer Methods,* 1:333–337.
114. Khorram, E., Khaledian, K., Khaledyan, M. (2014), A numerical method for constructing the Pareto front of multi-objective optimization problems, *Journal of Computational and Applied Mathematics,* 261 (2014), 158-171.
115. Kim, N.T.B., Luc, D.T., (2000), Normal cones to a polyhedral convex set and generating efficient faces in linear multi-objective programming. *Acta Mathematica Vietnamica 25, 101.*
116. Klamroth, K., Tind, J., Wiecek, M.M, (2002), Unbiased approximation in multicriteria optimization.*Math. Methods Oper. Res. 56, 413–457* .
117. Lin, J.G.(1975), Three methods for determining Pareto-optimal solutions of multiple-objective problems. *In:Ho, Y.C., Mitter, S.K. (eds.) Large-Scale Systems, Plenum Press, New York.*
118. Luc, D.T., Phong, T.Q., Volle, M.,(2005), Scalarizing functions for generating the weakly efficient solution set in convex multi-objective problems. *SIAM J. Optim. 15,* 987–1001 .
119. L. Zadeh, (1963), Optimality and non-scalared-valued performance criteria.*IEEE Trans. Automatic Control, pp. 59–60.*
120. Marler RT, Arora JS (2004), Survey of multi-objective optimization methods for engineering. *Struct Multidisc Optim,* 26:369–395.
121. Meng, H., Zhang, X., Liu, S. (2005), New quality measures for multiobjective programming. *Springer, Lecture Notes in Comput. Sci. 3611*pp. 1044-1048.
122. Messac A (1996), Physical programming: effective optimization for computational design.*AIAA J,* 34(1):149–158.
123. Messac A, Ismail-Yahaya A, Mattson CA, (2003), The normalized normal constraint method for generating the Pareto frontier. *Struct Multidisc Optim,* 25:86–98.
124. Messac A, Mattson C.A, (2004), Normal constraint method with guarantee of even representation of complete Pareto frontier. *AIAA J,* 42:2101–2111.
125. Messac A, Mattson C.A,(2002), Generating well-distributed sets of Pareto points for engineering design using physical programming.*Optim Eng,* 3:431–450.
126. Mattson C.A, Messac A (2005), Concept selection using s-Pareto frontiers. *AIAA J* 41:1190–1204.
127. Miettinen, K.(1999), Nonlinear Multi-objective Optimization, *International Series in Operations Research and Management Science, Vol. 12, Kluwer Academic, Dordrecht.*
128. Pascoletti, A., Serafini, P., (1984), Scalarizing vector optimization problems.*J. Optim. Theory Appl.* 42, 499–524.

129. Pinter, J. D., Linder, D., and Chin, P. (2006), Global optimization toolbox for maple: An introduction with illustrative applications.*Optimisation Methods and Software*21, 4 (2006), 565-582..
130. Ruzika, S., Wiecek, M.M., (2005), Approximation methods in multi-objective programming.*J. Optim. Theory Appl.* 126, 473–501.
131. Suppapitnarm A et al, (1999), Design by multi-objective optimization using simulated annealing. *International conference on engineering design ICED 99, Munich, Germany.*
132. Shukla, P. K. (2007), On the normal boundary intersection method for generation of efficient front. *In Computational Science-ICCS. Springer, pp. 310-317.*
133. Stewart, T.J. van den Honert R.C. (eds.), (1997), Trends in Multicriteria Decision Making.*Lectures Notes in Economics and Mathematical Systems 465. Springer, Berlin.*
134. Steuer, R.E., (1986), Multiple-Criteria Optimization: Theory, Computation, and Application.*John Wiley, New York.*
135. Siddiqui, S. Azarm, S. Gabriel, S.A., (2012), On improving normal boundary intersection method for generation of Pareto frontier, *Struct Multidisc Optim, DOI 10.1007/s00158-012-0797-1. Springer-Verlag.*
136. Sawaragi, Y. Nakayama, H. and T. Tanino, (1985), Theory of Multi-objective Optimization.*Academic Press, Orlando, FL.*
137. Santana-Quintero, L.V., Hernandez-Diaz, A.G., Molina, J., Coello Coello, C.A. and R. Caballero. DEMORS, (2010), A hybrid multi-objective optimization algorithm using differential evolution and rough set theory for constrained problems.*Computers & Operations Research,* 37(3):470-480.
138. Wang,Y.-N., Wu, L.-H. and X.-F. Yuan, (2010), Multi-objective self-adaptive differential evolution with elitist archive and crowding entropy-based diversity measure.*Soft Computing,* 14(3):193-209.
139. Yu, P.L., (1985), Multiple-Criteria Decision Making: Concepts, Techniques and Extensions. *Plenum Press, New York.*
140. Zhang, Q., Zhou, A., Zhao, S., Suganthan, P. N., Liu, W., and Ti- wari, S. (2008), Multiobjective optimization test instances for the cec 2009 special session and competition. *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*1-30.
141. Zhao, S.Z. and P.N. Suganthan, (2010), Multi-objective evolutionary algorithm with ensemble of external archives. *Int. J. of Innovative Computing, Information and Control,* 6(1):1713-1726.
142. Zitzler, E. and L. Thiele, (1999), Multi-objective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation,* 3(4):257-271.
143. Zitzler, E., K.Deb, and L.Thiele. (2000), Comparison of multiobjective evolutionary algorithms: empirical results. *Evol. Comput.J. 8,* 125-148..
144. Zopounidis, C. and P.M. Pardalos, (1995), Handbook of Multicriteria Analysis. *Springer, Berlin.*
145. Rodenas, R. G., Lopez, M. L., and Verastegui, D. (1999). Extensions of dinkelbach's algorithm for solving non-linear fractional programming problems. Top, 7(1):33–70.
146. Jaggi, M. (2013). Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In International Conference on Machine Learning, pages 427–435. PMLR.

147. Sivri, M., Albayrak, I., and Temelcan, G. (2018). A novel solution approach using linearization technique for nonlinear programming problems. International Journal of Computer Applications, 181(12):1–5.
148. Pint´er, J. D., Linder, D., and Chin, P. (2006). Global optimization toolbox for maple: An introduction with illustrative applications. Optimisation Methods and Software, 21(4):565–582.