# Mathematical Programming for Support Vector Machines

Doctoral Thesis

Belen Martin-Barragan

June, 2006

Supervisors:
Dr. Emilio Carrizosa
Dr. Dolores Romero Morales


UNIVERSIDAD D SEVILLA
**FACULTAD DE MATEMÁTICAS**
Dpto. de Estadística e Investigación Operativa

# ACKNOWLEDGEMENTS

Este trabajo ha estado financiado por

- la beca de Formación de Personal Docente e Investigador de la Junta de Andalucía.
- el proyecto BFM2002-04525-C02-02 y su continuación MTM2005-09362-C03-01 del Ministerio de Ciencia y Tecnología y el fondo FEDER de la Unión Europea.
- las ayudas a la investigación que anualmente la Junta de Andalucía ha concedido al grupo de investigación *Optimización*, FQM-329.
- el proyecto *Optimization Models for Data Mining*, de la fundación METEOR, Holanda.

This work has been financially supported by:

- A four-years pre-doctoral grant by Junta de Andalucía, Spain.
- The research projects BFM2002-04525-C02-02 and MTM2005-09362-C03-01 by Ministerio de Ciencia y Tecnología, Spain, and European Regional Development Fund (ERDF), European Union.
- The research group FQM-329 of Junta de Andalucía, Spain.
- The research project *Optimization Models for Data Mining*, granted by METEOR, The Netherlands.

En primer lugar debo agradecer a mis directores Loli y Emilio por la gran ilusión con que se involucraron, desde un principio, en la difícil tarea de dirigir, no sólo una tesis, sino todo lo necesario para mi formación científica. Los dos se han entregado al máximo a esta tarea y han hecho un gran esfuerzo, muchas veces revisándose el trabajo en aviones y aeropuertos, para que este trabajo saliera adelante.

Loli ha sido para mi, no sólo una estupenda directora de tesis, sino también un modelo al que imitar, un exigente (como es ella consigo misma) modelo al que sueño parecerme un día. Conocer a Emilio ha sido para mi una suerte excepcional. Pocos doctorandos habrán aprendido tanto de su director como yo de él. Además me ha inculcado valores que son importantes en un científico, como el espíritu crítico, la rigurosidad, el afán por aprender, la conciencia de que la investigación debe ser útil a la sociedad, ...

Me gustaría recordar aquí a la gente que he conocido durante estos años en congresos y estancias: muy especialmente a Lázaro Cánovas, a quien muchos compañeros guardaremos en el recuerdo. Un par de semanas en Ankara, en el Euro Summer Institute, hicieron que ahora tenga amigos repartidos por todo el mundo. Me encantaría volver a verlos, como a Ivonne, en alguno de mis próximos viajes. María, Aranxa, Emmy, Patrick y Ken hicieron más divertida y enriquecedora mi estancia en Oxford, así como Jan-Willem es el culpable de que tenga un recuerdo imborrable de la ciudad de Maastricht.

También quiero dar las gracias a los jóvenes científicos de precarios por su (nuestra) lucha por la dignificación laboral de la carrera investigadora desde su comienzo. Sobre todo muchos ánimos para Carola, y su ilusión porque la asociación precarios-Sevilla salga adelante.

En la vida hay amigos que vienen y van, muchos de ellos permanecen a nuestro lado aunque estén lejos o reaparecen cuando el contacto parecía perdido. Por ello quiero mencionar en estas líneas a Raquel, a Pedro, a Yolanda, a María del Mar, a Mariani, y a tantos otros amigos a los que aprecio. También a mis amigas de Cartaya, con las que solía compartir noches de verano antes de involucrarme en esta aventura de la investigación.

No es habitual en mí dar las gracias a mis padres, pero sin duda se las merecen. A mi madre quiero agradecer, además de la deliciosa comida que me prepara, que esté siempre pendiente de lo que necesito y se preocupe por mi salud, mi futuro, mi alimentación, ... mi vida en general. Cuando estoy lejos de ella nunca la echo de menos porque en mi mente nunca dejo de tenerla a mi lado. A mi padre quiero agradecerle su apoyo, su comprensión y su respeto a mis decisiones. De él he heredado el constante afán por aprender cosas nuevas. Espero que mi título de doctora le haga sentirse aún más orgulloso de mí.

Por último, darle las gracias a mi hermano, que siempre me apoya en los momentos difíciles.

En Sevilla, a 12 de Junio de 2006

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# 1. INTRODUCTION

## *1.1 Support Vector Machines: a bridge between Mathematical Programming and Data Mining*

During the last decade, the storage capacity of digital information has been duplicated every nine months. It grows, therefore, at a speed by far higher than the one anticipated by Moore's law for the growth of the computing power, [30, 43], causing the appearance of the so-called *data graves*, [30], data that are stored and rest peacefully, with nobody who demands them or remembers them.

The establishment of such data graves existence, and the consequent loss of opportunities for advance in business and knowledge, are causing an enormous interest to develop techniques that, complementary to the previously existing ones, allow to obtain unknown and potentially useful information from data of applications as diverse as Bioinformatics (genetic expression, ...), Customer Relationship Management (customer retention, market basket analysis, ...), bank (risk evaluation of credits, fraud detection, credit scoring, ...), Internet (webpage classification, spam filtering, ...), [1, 2, 3, 27, 33, 37, 38, 77].

We speak, using a common denomination in scientific publications, and, in particular, in the publishing lines of some of the journals with highest impact in our area of knowledge, of *Data Mining*. The references [2, 13, 38, 40, 76] can serve as introduction to the subject.

Examining, for example, the different options of the open-coded software `Weka`, [73], described in [76], it is observed that one of the pillars of Data Mining, although quite previous to this one, is the *classification*. We find, next to well-known procedures in the statistical community, like logistic regression, bayesian models, classification trees or neural networks, more recent ones, like the one we are dealing with in this work: the *Support Vector Machines* (SVM), that has jumped from the Statistical Learning

world, [21, 67, 68] to applications ... through Mathematical Programming. See [5, 9, 8, 18, 19, 50, 57, 61, 64, 78] for other classification methods that, like Support Vector Machines, use advanced techniques of Mathematical Programming.

In the classification problem, we have a set of objects $\Omega$, partitioned into $\sharp(\mathcal{C})$ classes (also called groups), where $\sharp(\cdot)$ denotes the cardinality of a set. The aim is to build a classification rule which predicts the class membership $c^u \in \mathcal{C}$ of an object $u \in \Omega$, by means of its *predictor vector* $x^u$. The predictor vector $x^u$ takes values in a set $X$ which is usually assumed to be a subset of $\mathbb{R}^p$, and the components $x_\ell$, $\ell = 1, 2, \ldots, p$, of the predictor vector $x$ are called *predictor variables*.

Not all the information about the objects in $\Omega$ is available, but only in a subset $I$, called the *training sample*, where both predictor vector and class membership of the objects are known. With this information, the classification rule must be built. For each $c \in \mathcal{C}$, denote by $I_c$ the set of objects of $I$ which belong to the class $c$, i.e., $I_c = \{u \in I : c^u = c\}$. In general, $\mathcal{C}$ is a finite set $\mathcal{C} = \{1, 2, \ldots, Q\}$. We assume throughout this thesis that each class is represented in the training sample, i.e., $I_c \neq \emptyset \forall c \in \mathcal{C}$. Special attention will be paid to the binary classification case, in which only two classes exists, $\mathcal{C} = \{-1, 1\}$. The most common approaches for the multigroup case consists on reducing the problem to a series of binary classification problems.

This introductory chapter is organized as follows. In Section 1.2, we expose the basic aspects of SVM for the binary classification case, and in Section 1.3, possible extensions to the multigroup case are presented. In some situations, SVM involves the optimization of large scale Linear Programs. In Section 1.4, the column generation technique, a well-known Mathematical Programming tool for solving that type of problems, is briefly described. The main aims of this thesis and an overview of the remaining chapters is included in Section 1.5.

## 1.2 Binary classification

We focus now on the case in which $\mathcal{C} = \{-1, 1\}$. SVM proposes a classification rule based on a score function $f$,

$$f(x) = \omega^\top x + \beta, \tag{1.1}$$

with $\omega \in \mathbb{R}^p$, and $\beta \in \mathbb{R}$. With this score function, the *linear classification rule* allocates those $x \in \mathbb{R}^p$ with $f(x) > 0$ to group 1, and those $x$ with $f(x) < 0$ to group $-1$. In case of ties, i.e. $f(x) = 0$, objects can be allocated randomly or by some predefined order. Throughout this thesis, following a worst case approach, ties will be considered as misclassifications.

The first question we may ask is if there exists or not $(\omega, \beta)$ such that the corresponding linear classification rule correctly classifies all objects in $I$.

**Definition 1.1:** A score function $f$ with coefficients $(\omega, \beta)$ is said to *separate* the training sample $I$ if

$$c^u \left(\omega^\top x^u + \beta\right) > 0 \quad \forall u \in I. \tag{1.2}$$

In addition, the training sample $I$ is called *separable* if there exists a score function with coefficients $(\omega, \beta)$ that separates $I$. Otherwise, $I$ is said to be nonseparable.

In Section 1.2.1, we deal with the separable case, in which $I$ is assumed to be separable, describing the so-called hard-margin approach. In Section 1.2.2, for the nonseparable case, the hard-margin approach is used after a transformation on the data. A different approach for the nonseparable case, is described in Section 1.2.3, which is useful to avoid overfitting, phenomenon which happens when a low misclassification rate in $I$ does not generalize to forthcoming objects. Other alternatives for the nonseparable case are presented in Section 1.2.4.

### *1.2.1 The hard-margin approach*

Any solution $(\omega, \beta)$ of (1.2) satisfies that $\omega \neq 0$. In particular, $(\omega, \beta)$ generates a hyperplane, $\{x \in \mathbb{R}^p : \omega^\top x + \beta = 0\}$, such that all objects in the halfspace $\{x \in \mathbb{R}^p : \omega^\top x + \beta > 0\}$ will be allocated to class 1, whereas all the objects in the halfspace $\{x \in \mathbb{R}^p : \omega^\top x + \beta < 0\}$ will be allocated to class -1.

When $I$ is separable, the system (1.2) has infinite solutions, that generate infinite different hyperplanes. How could we choose one of these solutions? The classification quality *in the training sample* is identical for all those generated classification rules: all correctly classify every object in $I$. However, not all of them seem equally reasonable. In Figure 1.1 we can see two hyperplanes, both separating $I$ (classes are represented by circles and squares). Very intuitively, we can think that the hyperplane represented by the thick

line is more convenient than the one represented by the thin line. In particular, the former allocates the object represented with '?' to the square class, when it seems much more likely to belong to the circle class.

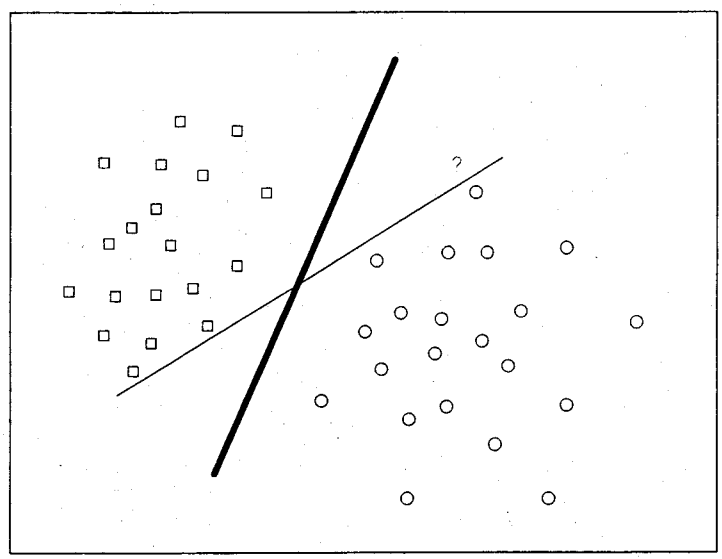


*Fig. 1.1:* Two rules that classify equally well?

The previous example intuitively indicates the convenience of choosing a hyperplane that is *far away* from both classes. SVM is based on this principle, as described in what follows. Given a fixed norm $\|\cdot\|$ in $\mathbb{R}^p$ to measure the distances (for instance, the Euclidean one), the distance between $x^u$, for an object $u \in I$, and the halfspace where it is misclassified is given by

$$\rho^u(\omega, \beta) = \max\left\{\frac{c^u(\omega^\top x^u + \beta)}{\|\omega\|^\circ}, 0\right\},$$

where $\|\cdot\|^\circ$ denotes the dual norm of $\|\cdot\|$, see [12].

**Definition 1.2:** Define the *margin of object* $u \in I$ by $\rho^u(\omega, \beta)$. We call the *margin on the training sample* $I$ the minimum $\rho^u$ :

$$\rho^I(\omega, \beta) = \min_{u \in I} \rho^u(\omega, \beta).$$

It has been shown, [67, 68], that the probability of misclassifying a forthcoming individual is bounded by an amount that depends on the margin. This provides a theoretical foundation for choosing the hyperplane which is furthest away from both classes. Thus, the sought classifier is the one that not only correctly classifies all the objects of $I$, but it also has maximum margin:

$$\begin{array}{lll} \max & \rho^I(\omega,\beta) & \\ \text{s.t.:} & c^u\left(\omega^\top x^u+\beta\right)>0 & \forall u\in I, \\ & \omega\in I\!R^p,\ \beta\in I\!R. & \end{array} \tag{1.3}$$

This problem is called the hard-margin maximization problem, as opposed to the soft-margin version, explained in Section 1.2.3, where some objects in $I$ are allowed to be misclassified.

Geometrically, the search of the classifier of maximum margin can be seen as the problem of finding the band of maximal width (the distances measured with norm $\|\cdot\|$) that leaves a group to each side, as can be seen in Figures 1.2-1.3 for the norms $L_2$ and $L_\infty$.

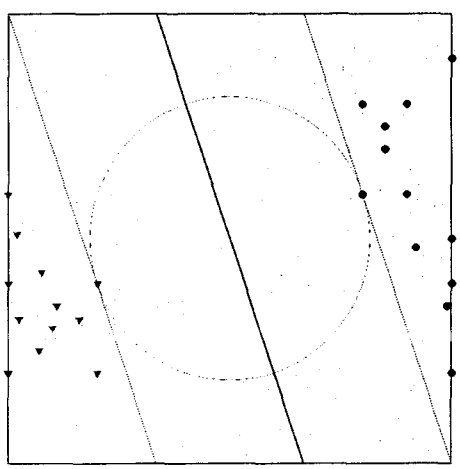

*Fig. 1.2:* Maximal margin ($L_2$ norm)

Note that, for $\mu>0$, $(\mu\omega,\mu\beta)$ and $(\omega,\beta)$ yield the same classification rule, in the sense that the classification of any object is exactly the same by both

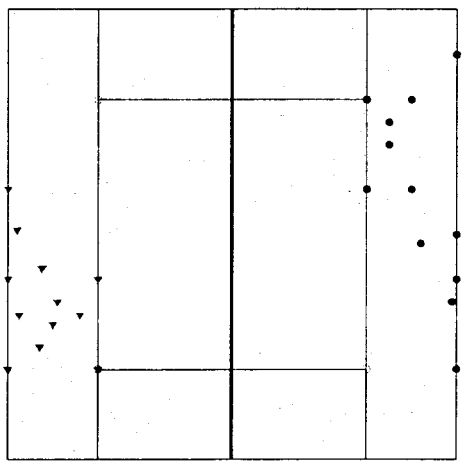

*Fig. 1.3:* Maximal margin ($L_\infty$ norm)

rules. Moreover, by Definition 1.2, $\rho^I(\mu\omega, \mu\beta) = \rho^I(\omega, \beta)$. Using this homogeneity property of the margin function, the margin maximization problem can be formulated as the following convex problem with linear constraints:

$$\begin{array}{ll} \min & \|\omega\|^\circ \\ \text{s.t.:} & c^u \left(\omega^\top x^u + \beta\right) \geq 1 \ \forall u \in I \\ & \omega \in I\!R^p, \ \beta \in I\!R. \end{array} \tag{1.4}$$

If, to measure distances, we have used, like in Figure 1.3, a polyhedral norm $\|\cdot\|$ (i.e., whose unit ball is a polyhedron), then its dual $\|\cdot\|^\circ$ is also polyhedral, and therefore Problem (1.4) can be reformulated as a Linear Programming problem, see for instance [49, 56], resoluble with commercial Linear Programming solvers such as CPLEX, [42]. However, if the problem is very large, as usually happens in Data Mining applications, there exist advanced Linear Programming techniques, such as column generation, briefly described in Section 1.4. In addition, for databases in which the number of predictor variables $p$ is large, the use of polyhedral norms yields solutions where many of the components of the vector $\omega$ are null, corresponding to classifiers that do not use all the predictor variables available. Such predictor variables can thus be discarded, with the advantage of obtaining cheaper or more interpretable classifiers. The most studied case in the literature, is not,

nevertheless, the one that has $\|\cdot\|$ as a polyhedral norm, but the Euclidean one, where Problem (1.4) is equivalent to the following convex quadratic problem with linear constraints:

$$\begin{array}{ll} \min & \omega^\top \omega \\ \text{s.t.:} & c^u\left(\omega^\top x^u + \beta\right) \geq 1 \ \forall u \in I \\ & \omega \in I\!R^p,\ \beta \in I\!R. \end{array}$$

In [56], empirical results show that 'in terms of separation performance, $L_1$, $L_\infty$ and Euclidean norm-based SVMs tend to be quite similar'.

### 1.2.2 Embedding into a feature space

In Section 1.2.1 we have assumed that $I$ is separable. If this is not the case, Problem (1.4) is infeasible, thus alternative approaches must be applied. One of these approaches consists of applying to the data, as a preprocessing step, a mapping $\Phi : I\!R^p \to E$, where $E$ is, in principle, a subset of $I\!R^N$ for a large $N$, and such that, in the new space, the transformed training sample $\hat{I} = \{(\Phi(x^u), c^u) : u \in I\}$ is separable, [14, 24, 25, 29, 41], and the hard-margin approach is applied there. After that, $\omega \in I\!R^N$, $\beta \in I\!R$, are sought and the classification rule is built, based on the score function $f$,

$$f(x) = \omega^\top \Phi(x) + \beta, \tag{1.5}$$

allocating $u$, as in the separable case, to group 1 if $f(x) > 0$, and to group $-1$ if $f(x) < 0$.

This rule is linear in the transformed data, but nonlinear in the original space $I\!R^p$. The different components of $\Phi$ are usually called *features*, whereas the space $E$ of the transformed data is called *feature space*.

Rephrasing Problem (1.4), but in the feature space $E$ instead of the space $X \subset I\!R^p$, the following margin maximization problem is obtained.

$$\begin{array}{ll} \min & \|\omega\|^\circ \\ \text{s.t.:} & c^u\left(\omega^\top \Phi(x^u) + \beta\right) \geq 1 \ \forall u \in I \\ & \omega \in I\!R^N,\ \beta \in I\!R. \end{array} \tag{1.6}$$

For the case in which the norm $\|\cdot\|$ is polyhedral and $E$ has a high dimension (but finite), Problem (1.6) is formulated as a large scale Linear Programming problem, for which the column generation technique, briefly described in Section 1.4, is specially advisable. For example, it could be

useful for automatic selection of the most influencing predictor variables [16] and interactions [17]. This is the approach followed in Chapters 2 and 3. As said before, the choice of a polyhedral norm contributes to get a solution of Problem (1.6) with many null components, which would indicate that many of the features $\phi$ are not used by the classifier.

If the Euclidean norm is used to measure distances in the feature space, Problem (1.6) becomes a quadratic convex problem whose dual is

$$\begin{array}{ll} \max & \sum_{u\in I}\lambda^u - \frac{1}{2}\sum_{u,v\in I}\lambda^u\lambda^v c^u c^v \Phi(x^u)^\top \Phi(x^v) \\ \text{s.t.:} & \sum_{u\in I} c^u\lambda^u = 0 \\ & \lambda^u \geq 0 \qquad\qquad \forall u \in I. \end{array} \tag{1.7}$$

Defining the *kernel* as $K : (x, y) \in I\!R^p \times I\!R^p \longrightarrow \Phi(x)^\top\Phi(y) \in I\!R$, Problem (1.7) becomes

$$\begin{array}{ll} \max & \sum_{u\in I}\lambda^u - \frac{1}{2}\sum_{u,v\in I}\lambda^u\lambda^v c^u c^v K(x^u, x^v) \\ \text{s.t.:} & \sum_{u\in I} c^u\lambda^u = 0 \\ & \lambda^u \geq 0 \qquad\qquad \forall u \in I. \end{array} \tag{1.8}$$

In order to solve Problem (1.8), it is not even needed the explicit knowledge of $\Phi$, but only an algorithm to evaluate the kernel $K$ induced by $\Phi$. In this way, the feature space does not need to be a subset of $I\!R^N$, and could be a general Hilbert space [23].

The maximization problem is quadratic concave, with as many decision variables as elements in $I$, and with only one constraint, linear, in addition to the nonnegativity ones. The dimension of this problem is, therefore, independent of the dimension $p$ of the original data and the dimension of the feature space $E$. Hence, Problem (1.8) is a specially attractive formulation in applications with not too many objects, but high dimensional, like the problems found, for example, in [27, 77]. For more details, see, for example [23, 41].

### 1.2.3 *The soft-margin approach*

An alternative strategy (and sometimes complementary to the later) to deal with the nonseparable case, is the one based on the soft-margin maximization, [21, 23, 41], in which, starting off with the infeasible Problem (1.4), their constraints are perturbed to make it feasible, introducing a penalty in the

objective to control the deviations. In general, Problem (1.4) is formulated as

$$\begin{array}{ll} \min & \|\omega\|^\circ + C\|\xi\|^* \\ \text{s.t.:} & c^u\left(\omega^\top x^u + \beta\right) + \xi^u \geq 1, \qquad \forall u \in I \\ & \omega \in I\!R^p,\ \beta \in I\!R,\ \xi \in I\!R^{\sharp(I)}, \end{array} \tag{1.9}$$

where $\|\cdot\|^*$ is a given norm, that does not need to be the same as the norm $\|\cdot\|^\circ$ used for the coefficients $\omega$, and $C$ is a constant, used to balance the deviations $\xi$ and the margin of the correctly classified objects, usually chosen by crossvalidation techniques, [47].

For example, using the polyhedral norm $\|\cdot\|_1$ for both, $\omega$ and the deviations $\xi = (\xi_u)_{u\in I}$, the relaxed Problem (1.9) is formulated as

$$\begin{array}{ll} \min & \|\omega\|_1 + C\|\xi\|_1 \\ \text{s.t.:} & c^u\left(\omega^\top x^u + \beta\right) + \xi^u \geq 1, \qquad \forall u \in I \\ & \omega \in I\!R^p,\ \beta \in I\!R,\ \xi \in I\!R^{\sharp(I)}. \end{array} \tag{1.10}$$

Other choices for $\|\cdot\|^*$ yield different soft-margin problems and the one that best fits to the model can be used.

Sometimes, the soft-margin approach is used even for a separable $I$, because that approach has been empirically shown to avoid overfitting.

Both approaches, an embedding $\Phi$ and the soft-margin maximization, can be used together, yielding, for norm $\|\cdot\|_1$, the following problem:

$$\begin{array}{ll} \min & \|\omega\|_1 + C\|\xi\|_1 \\ \text{s.t.:} & c^u\left(\omega^\top \Phi(x^u) + \beta\right) + \xi^u \geq 1, \qquad \forall u \in I \\ & \omega \in I\!R^N,\ \beta \in I\!R,\ \xi \in I\!R^{\sharp(I)}, \end{array} \tag{1.11}$$

where $N$ denotes the cardinality of the set of features $\mathcal{F}$ which define the embedding $\Phi = (\phi)_{\phi\in\mathcal{F}}$. Problem (1.11) can be formulated as the following Linear Program

$$\begin{array}{lll} \min & \sum_{\phi\in\mathcal{F}}(\omega_\phi^+ + \omega_\phi^-) + C\sum_{u\in I}\xi^u & \\ \text{s.t.:} & \sum_{\phi\in\mathcal{F}}(\omega_\phi^+ - \omega_\phi^-)c^u\phi(x^u) + \beta c^u + \xi^u \geq 1 & \forall u \in I \\ & \omega_\phi^+ \geq 0 & \phi \in \mathcal{F} \\ & \omega_\phi^- \geq 0 & \phi \in \mathcal{F} \\ & \xi^u \geq 0 & \forall u \in I \\ & \beta \in I\!R. & \end{array} \tag{1.12}$$

### 1.2.4 Other alternatives

Recently, other alternative strategy for nonseparability has been proposed: instead of transforming the problem into another one on which $I$ is separable (and where, for instance, the maximin criterion is applicable), a different criterion can be used that does not require separability. Such is the case, for example, of the method described in [51, 59], where the sum of the distances between each misclassified point and the halfspace where it is correctly classified, is minimized. This yields the following optimization problem,

$$\begin{array}{ll} \min & \sum_{u\in I} \max\left\{\frac{-c^u(\omega^\top x^u+\beta)}{\|\omega\|^\circ}, 0\right\} \\ \text{s.t.:} & \omega \in I\!R^p,\ \beta \in I\!R, \end{array}$$

which enjoys the good properties of the median hyperplane, [52, 58], but, like that one, is multimodal.

## 1.3 Multigroup classification

The extension of SVM for the case in which the number of groups $Q$ is greater than two is not trivial. The most used approaches are actually based on combining the classifiers obtained as solutions of several two-classes SVMs. Already in [21], which is the origin of SVM, the method *one-against-all* is set out. There, $Q$ classification rules are built. For $c \in \mathcal{C}$, the $c$-th classification rule is built by means of SVM applied to the classification of objects of class $c$ as opposed to the rest of the objects of $I$. The final classification rule allocates an object to the class to which it is allocated the highest number of times among the built classification rules.

Another approach based on the same idea is the so-called *one-against-one* [39], in which SVM is applied to each pair of classes, ignoring the remaining objects. The $\frac{Q(Q-1)}{2}$ obtained classifiers are combined, for example, as in the one-against-all approach. A different way to combine these classification rules can be seen in [60].

Other authors, as for example in [10, 34, 75], have proposed the use of a *(multiple) score function* $f = (f_c)_{c\in\mathcal{C}}$ with $Q$ components $f_c : \mathcal{X} \longrightarrow I\!R$ of the form (1.1). Then, an object $u \in \Omega$ will be allocated to the class $c^*$ with greater score function value.

$$c^* = \arg\max_{c \in C} f_c(z).$$

As in the two-group case, in case of ties, the object will be unclassified by this rule, and can be later allocated randomly or by a prefixed order to some class in $\arg\max_{c \in C} f_c(z)$. Following a worst-case approach, we will consider those objects as misclassified throughout this thesis.

Special attention is paid to this multiple score function approach, since it is considered in Chapter 5. Every component of the multiple score function $f$ is a score function $f_c$ with the form (1.1), used in Section 1.2. Denote $W = (\omega^1, \ldots, \omega^Q)$ and $b = (\beta^1, \ldots, \beta^Q)$, where $\omega^c \in \mathbb{R}^p$, and $\beta^c \in \mathbb{R}$ are the coefficients of the score function $f_c$ :

$$f_c(x) = \sum_{k=1}^{p} \omega_k^c x_k + \beta^c. \tag{1.13}$$

We now extend the concept of separability given in Section 1.2 to the multi-group case.

**Definition 1.3:** A score function $f = (f_c)_{c \in C}$ with the form (1.13) is said to *separate* $I$ if

$$f_{c^u}(x^u) > f_j(x^u) \quad \forall j \neq c^u, \forall u \in I.$$

Moreover, $I$ is said to be *separable* if there exists a score function $f = (f_c)_{c \in C}$, with the form (1.13) separating $I$.

Notice that, for binary classification, $C = \{-1, 1\}$, this concept of separability is equivalent to the concept given in Section 1.2, Definition 1.1, as we see in the following property.

**Property 1.4:** Let $C = \{-1, 1\}$ and let $I$ be a training sample. The two following conditions are equivalent:

- There exists a multiple score function $\hat{f} = (f_{-1}, f_1)$ separating $I$.
- There exists a score function $f$ of the form (1.1) separating $I$.

**Proof.** Given a multiple score function $\hat{f} = (f_{-1}, f_1)$ with coefficients $(\omega^{-1}, \beta^{-1})$ and $(\omega^1, \beta^1)$, the score function $f$ defined by the coefficients $\omega = \omega^1 - \omega^{-1}$ and $\beta = \beta^1 - \beta^{-1}$ separates $I$.

Conversely, given $(\omega, \beta)$, satisfying (1.2), setting $\omega^1 = \omega$, $\beta^1 = \beta$, $\omega^{-1} = 0$ and $\beta^{-1} = 0$, we have a multiple score function that separates $I$. □

As in Section 1.2, we first explore the hard-margin formulations for the separable case (Section 1.3.1) and later, in Section 1.3.2, we extend it to the soft-margin approach, in which some objects are allowed to be misclassified.

### 1.3.1 The hard-margin approach

Suppose that $I$ is separable, which ensures the existence of at least one score function $f$ separating $I$. Uniqueness of such a score function $f$, separating $I$, never holds. Indeed, it is easy to see that given $(\hat{\omega}, \hat{\beta}) \in \mathbb{R}^{p+1}$ the classification rules obtained by $(W, b)$ and $(\tilde{W}, \tilde{b})$ with $\tilde{\omega}^c = \lambda\omega^c + \hat{\omega}$ and $\tilde{\beta}^c = \lambda\beta^c + \hat{\beta}$, for all $c \in \mathcal{C}$, are equivalent for all $\lambda > 0$, in the sense that both allocate objects to the same classes.

Moreover, there are also more than one score function that separate $I$ and they are not equivalent. For instance, given a score function separating $I$, let $\varepsilon$ be any value satisfying:

$$0 < \varepsilon < \min_{u \in I} \min_{j \neq c^u} \{f_{c^u}(x^u) - f_j(x^u)\}.$$

The function $f^\varepsilon = (f_1 + \varepsilon, f_2, \ldots, f_Q)$ also separates $I$. We need a criterion for choosing one of these multiple score functions. In the binary classification case, following Vapnik's publications in generalization ability [67], the margin maximization criterion was presented. Now we present an extension of the concept of margin for the multigroup case.

**Definition 1.5:** Let $\|\cdot\|^\circ$ be a norm in $\mathbb{R}^{pQ}$. The *margin of an object* $u$ with respect to the score function $(W, b)$, with $W \neq 0$, is defined as

$$\rho^u(W, b) = \min_{j \in \mathcal{C}, j \neq c^u} \frac{f_{c^u}(x^u) - f_j(x^u)}{\|W\|^\circ}.$$

The *margin of a score function* $(W, b)$ with respect to the training sample $I$ is the minimum:

$$\rho^I(W, b) = \min_{u \in I} \rho^u(W, b).$$

As said in the binary case, a common choice for the norm $\|\cdot\|^\circ$, is the Euclidean norm, but other norms might be useful. For example, in Chapter

5, we use the $L_1$ norm, which allows us to formulate the maximal margin problem as a Linear Program, solvable with existing commercial software.

Now, we consider the problem of maximizing the margin

$$\begin{array}{ll} \max & \rho^I(W,b) \\ \text{s.t:} & (W,b) \in I\!R^{pQ} \times I\!R^Q. \end{array} \tag{1.14}$$

Denote by $\theta(W,b)$ the amount

$$\min_{u \in I} \min_{j \in C, j \neq c^u} \left( f_{c^u}(x^u) - f_j(x^u) \right),$$

which is the numerator of the margin $\rho^I(W,b)$. Note that for all $\lambda > 0$, solutions $(W,b)$ and $(\lambda W, \lambda b)$ yield the same classification rule, and, following Definition 1.5, they also have the same margin. Using this property, Problem (1.14) can be formulated as the following convex probblem:

$$\begin{array}{ll} \max & \theta(W,b) \\ \text{s.t:} & \|W\|^\circ \leq 1, \\ & (W,b) \in I\!R^{pQ} \times I\!R^Q, \end{array} \tag{1.15}$$

in the sense that any optimal solution of Problem (1.15) is also optimal for Problem (1.14), and for any optimal solution $(W^*, b^*)$ of Problem (1.14),

$$(\hat{W}, \hat{b}) = \frac{1}{\|W^*\|^\circ}(W^*, b^*)$$

is an optimal solution of (1.15).

Formulation (1.15) is derived by using the normalization constraint $\|W\|^\circ = 1$, i.e. normalizing the denominator of $\rho^I(W,b)$ and maximizing its numerator, $\theta(W,b)$. Another equivalent formulation can be suggested, normalizing the numerator and minimizing $\|W\|^\circ$, yielding

$$\begin{array}{ll} \min & \|W\|^\circ \\ \text{s.t.:} & \theta(W,b) \geq 1 \\ & (W,b) \in I\!R^{pQ} \times I\!R^Q. \end{array} \tag{1.16}$$

Throughout the thesis, we will use one or another approach in order to obtain formulations with better properties.

### 1.3.2 The nonseparable case

Two different approaches are explained, in Sections 1.2.2 and 1.2.3, for the nonseparable binary classification: embedding the data into a higher dimensional space, and the soft-margin approach, which allows some objects to be misclassified. Both approaches can be extended to the nonseparable multigroup case.

After applying an embedding $\Phi : I\!R^p \to E$, with $\Phi = (\phi_1, \phi_2, \ldots, \phi_N)$ to the data, the score function $f$ can be expressed as

$$f_c(x) = \sum_{k=1}^{N} \omega_k^c \phi_k(x) + \beta^c = (\omega^c)^\top \Phi(x) + \beta^c, \tag{1.17}$$

with $\omega^c \in I\!R^N$, and $\beta^c \in I\!R$, $\forall c \in \mathcal{C}$.

As in the binary classification case, a soft-margin version of Problem (1.14) can be developed, hopefully avoiding overfitting by allowing some objects to be misclassified. A score function $f$ which does not separate $I$, will be infeasible for Problem (1.16) because constraint $\theta(W, b) \geq 1$ does not hold. Such constraint can be rephrased as the set of constraints

$$(\omega^{c^u})^\top \Phi(x^u) + \beta^{c^u} - (\omega^j)^\top \Phi(x^u) - \beta^j \geq 1, \quad \forall u \in I, \forall j \in \mathcal{C}, j \neq c^u.$$

Hence, in order to allow a score function $f$ that does not separate $I$, we relax these constraints by adding perturbations $\xi_u^j$, which are later penalized in the objective function. Let $\xi$ be the vector of all the perturbations $\xi_u^j$. Then the soft-margin version of Problem (1.16) can be formulated as:

$$\begin{array}{ll} \min & \|W\|^\circ + C\|\xi\|^* \\ \text{s.t.:} & (\omega^{c^u})^\top \Phi(x^u) + \beta^{c^u} - (\omega^j)^\top \Phi(x^u) - \beta^j + \xi_u^j \geq 1, \\ & \qquad\qquad \forall u \in I, \forall j \in \mathcal{C}, j \neq c^u \\ & (W, b) \in I\!R^{NQ} \times I\!R^Q, \end{array} \tag{1.18}$$

where $\|\cdot\|^*$ is a given norm, that does not need to be the same as the norm $\|\cdot\|$ used for the coefficients $W$, and $C$ is a given constant whose purpose is to trade off the perturbations $\xi_u^j$ and the margin. A popular choice for the norm $\|\cdot\|^*$ is, for instance, the $L_1$ norm, $\|\xi\|_1 = \sum_{u \in I, j \in \mathcal{C}, j \neq c^u} \xi_u^j$, [75].

In principle, formulation (1.18) uses, for each object $u \in I$, $Q - 1$ perturbations, one for every constraint. For a given score function defined by $(W, b)$, if an object $u$ is missclassified, the amount

$$f_{c^u}(x^u) - f_j(x^u) = (\omega^{c^u})^\top \Phi(x^u) + \beta^{c^u} - (\omega^j)^\top \Phi(x^u) - \beta^j$$

can be seen as a measure of how far object $u$ is from being correctly classified. Hence, it could seem more logical to use just one perturbation $\eta_u$ per object $u$. This can be done, within the framework of formulation (1.18), via the choice of norm $\|\xi\|^* = \sum_{u\in I} \max_{j\in\mathcal{C}, j\neq c^u} \xi_u^j$, yielding the following problem:

$$\begin{array}{ll} \min & \|W\|^\circ + C\|\eta\|_1 \\ \text{s.t.:} & (\omega^{c^u})^\top \Phi(x^u) + \beta^{c^u} - (\omega^j)^\top \Phi(x^u) - \beta^j + \eta_u \geq 1, \\ & \qquad\qquad \forall u \in I, \forall j \in \mathcal{C}, j \neq c^u \\ & (W, b) \in \mathbb{R}^{NQ} \times \mathbb{R}^Q, \end{array} \tag{1.19}$$

where $\eta$ denotes the vector $(\eta_u)_{u\in I}$ of one deviation per object.

## 1.4 Large scale linear programs

When the embedding $\Phi$ has a high (finite) dimension, Problem (1.12) becomes a large scale Linear Program, for which we propose to use the well-known Mathematical Programming tool called Column Generation, initially introduced for the cutting-stock problem [32]. In this section, we briefly describe such a tool, that will be used in Chapters 2 and 3.

When a Linear Programming problem (P) has a high number of decision variables, instead of solving it directly, the Column Generation technique solves a series of reduced problems where only a subset $V$ of the set of decision variables $\mathcal{V}$ is considered. Decision variables are iteratively added to $V$ as needed.

For $V \subset \mathcal{V}$, let *Master Problem* (P-$V$) be Problem (P), we are interested in solving, where all the decision variables not in $V$ are set to zero. We start with an initial subset $V$ of decision variables, for example, a random sample from $\mathcal{V}$. Once the initial set is generated, we solve Problem (P-$V$). The next step is to check whether the current solution is optimal for Problem (P) or not, and, in the latter case, look for a new decision variable $v$ in $\mathcal{V} \setminus V$ such that the solution of the new Problem (P-$(V \cup \{v\})$) is better than the one of Problem (P). Then, the decision variable $v$ is added to subset $V$, and Problem (P-$V$) is solved again. This process is repeated until no other promising decision variable is found. A simple summary scheme of the column generation algorithm can be seen below.

In each step, we ideally would like to find which decision variable is the most promising, in the sense that, adding it to $V$ yields the maximal improvement to the objective function of Problem (P). The problem of finding

the most promising decision variable, called the pricing problem, is related to the problem of finding the most violated constraint in the dual formulation of Problem (P). Sometimes, in order to accelerate the column generation algorithm, several good decision variables, obtained by the pricing problem are added to $V$ instead of only one.

**CG-summary: Summary of the column generation algorithm**

**Step 0.** Get an initial set of decision variables $V$.

**Step 1.** Solve the pricing problem to generate a new decision variable $v$.

**Step 2.** If no new promising decision variable are found, then STOP: we have found the optimal solution of Problem (P). Otherwise, add it (them) to the set $V$, solve Problem (P-$V$), and go to Step 1.

When, instead of using an exact algorithm to solve the pricing problem, a heuristic is used, then the column generation algorithm yields a good solution of Problem (P), but with no guarantee of optimality.

## *1.5 Thesis overview*

Building classification rules based on margin maximization has shown to be very efficient in Data Mining applied fields. Despite of the great advances obtained in the last years, there are still many aspects (modeling, numerical and algorithmic issues) to explore.

In this thesis we present some proposals in which Mathematical Programming tools are used to obtain classifiers with some interesting properties. In practical applications, the main goal is to obtain classifiers with a low percentage of misclassified objects, but the fact that, on top of that, they are also easily interpretable, or cheap, or useful to detect relevant predictor variables and their interactions, might also be of great interest. For instance, in microarray analysis, interpretability is one of the issues that influences the choice of a prediction method, [63], where easily interpretable models, which might help to provide new medical insights, are sometimes preferred. In some fields, as diverse as cancer diagnosis and credit scoring, doctors or lenders might find important to easily explain the classification rule and detecting which combinations of predictor variables are critical to predict class membership. Sometimes, the importance or cost of correctly classifying an object

varies depending on the class it belongs to. In this way, methods taking into account this importance or cost of misclassification depending on the class are highly desirable in many applications.

Our main aims in this work are:

1. Model, with SVM techniques, classification problems that incorporate costs (misclassification costs, feature costs, measurement costs).

2. Automatically detect adequate nonlinear data transformation and relevant interactions between variables for SVM.

3. Analyze the empirical behavior of the proposed methods in real databases standard in Data Mining literature.

In Chapter 2, based on our paper [16], an SVM-based model is proposed, which, while enjoying a good classification rate, automatically detects the most important predictor variables, and those values which are critical for the classification. The method involves the optimization of a large scale Linear Programming problem, for which we use the well-known Column Generation technique described in Section 1.4. Moreover, the proposed classifier is robust against the presence of outliers. In Chapter 3 we extend this method to one that, apart from detecting the relevant predictor variables, also detects the most relevant interactions between them. The classification ability of the proposed method is comparable to standard SVM for different kernels and clearly better than Classification Trees. These results are the basis of our paper [17].

The problem of incorporating misclassification costs that depend on the classes is analyzed in Chapter 4. Based on our reference [14], in that chapter we propose a model in which, for a score function $f$, the margin of a class $c$ is defined, independently of the margin of the other class, and the problem of simultaneously maximizing both margins is analyzed.

In many pracical applications it is important that the built classifier is cheap or quick to apply it to new objects. For instance, in a poll by KDnuggets, [45], where both practitioners and academics participated, 'dealing with unbalanced and cost-sensitive data' was selected between the most important Data Mining topics, where cost-sensitive data means data 'which has different cost to get it', [46]. This situation is illustrated in [46] with the following example:

*E.g. for a medical diagnosis, we can use data from a blood test or a spinal fluid test, but blood test is much cheaper (and easier) to get than a spinal fluid test. Making decision in such cases (which is really all the real-world cases) requires combining accuracy and other metrics with the cost of getting the data.*

In Chapter 5, based on our paper [15], we propose, for multi-group classification, a biobjective optimization model which takes into account both measurement costs (economical costs, risk incurred by the measuring process, computational times, space requirements), and misclassification rate. Again, within an SVM framework, misclassification rate is taken into account via margin maximization.

We introduce a biobjective mixed integer problem, for which Pareto optimal solutions are obtained. Those Pareto optimal solutions correspond to different classification rules, among which the user would choose the one yielding the most appropriate compromise between the cost and the expected misclassification rate.

Computational results show the great potential of Mathematical Programming tools, in particular, Linear and Mixed Integer Programming, to develop powerful variants of Support Vector Machines.

# 2. DETECTING RELEVANT VARIABLES

## *2.1 Introduction*

In practical applications, classification accuracy of the obtained classifier is not the only concern, but other characteristics of the classifier are also taken into account. For instance, in microarray analysis, interpretability is one of the issues that influences the choice of a prediction method, [63]. Classifiers obtained by SVM have shown to have good classification ability but are, in general, hard to interpret. In some application fields, practitioners, such as doctors or businessmen, may be very unwilling to use a classifier they cannot interpret. For them, Data Mining methods sometimes proceed like a black-box, so they would not feel confident enough to use the classifier unless they can interpret it somehow.

For instance, it is easy to interpret and manage queries of type

- Is predictor variable $\ell_1$ big?
- Is predictor variable $\ell_2$ small?
- Does predictor variable $\ell_3$ attain a very extreme value?

where the concept of 'big', 'small' and 'extreme value' must be quantified, e.g. in the form

$$\boxed{\text{is predictor variable } \ell \text{ greater than or equal to } b?} \tag{2.1}$$

This type of queries are used e.g. in Classification Trees. In this way, practitioners can interpret the classifier, describing how it works. Moreover, they can directly see which role the different predictor variables are playing in the classifier, and detect the values of a predictor variable critical for the classification.

In this chapter we propose a new model that, by using piecewise constant functions, automatically selects the adequate scale for each predictor

variable. The obtained rule is based on queries of type (2.1), which makes interpretability easier. Moreover, as shown empirically, the obtained classifiers are robust against the presence of outliers.

We restrict ourselves to the case in which two classes exist, $\mathcal{C} = \{-1, 1\}$. The multiclass case, described in Section 1.3, can be reduced to a series of two-class problems, as has been suggested e.g. in [39, 41, 68]. A summary of these strategies can be found in Section 1.3.

We work in an SVM-based framework, where the feature space is defined by binarizing each predictor variable, and considering all the possible cutoffs. For this reason, we call our method the Binarized Support Vector Machine, BSVM. Numerical results show that the proposed approach gives a classifier that behaves similar to the standard linear SVM and clearly better than Classification Trees. Moreover, the tradeoff between interpretability and classification ability can be controlled via an upper bound on the number of features allowed.

The classifier proposed in this chapter, BSVM, is described in Section 2.2. Since the number of features to be considered may be huge, the BSVM method yields an optimization problem with a large number of decision variables, namely $\sharp(I) \cdot p$, where $\sharp(\cdot)$ denotes the cardinality of a set. In Section 2.3, a Column-Generation-based algorithm, see Section 1.4 for more details in Column Generation techniques, is proposed in order to solve such an optimization problem. Numerical results are shown in Section 2.4, whereas conclusions are discussed in Section 2.5.

## 2.2 Binarized Support Vector Machines

In practical applications, simple rules of type (2.1) are very desirable because of their interpretability. For example, a doctor would say that having high blood pressure is a symptom of disease. Moreover, quantitative variables usually appear together with the qualitative ones. In order to deal with continuous variables in the form of the presence or absence of certain symptom, we propose the use of rules of type (2.1), where the cutoff value $b$ need to be chosen. Choosing the threshold $b$ from which a specific blood pressure would be considered high is not usually an easy task.

We theoretically consider all the possible rules of type (2.1), mathemati-

cally formalized by the function

$$\phi_{\ell b}(x) = \begin{cases} 1 & \text{if } x_\ell \geq b \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

for $b \in \mathbb{R}$ and $\ell = 1, 2, \ldots, p$. In our model, each $\phi_{\ell b}$ is a feature that defines a component of the transformation $\Phi$ used in the nonlinear score function (1.5).

This binarization procedure could be done in a tedious preprocessing step, where all the possible features are created. However, we do not do it as a preprocessing step but, as we will see later on, propose a method to generate features when needed.

The set of possible cutoff values $b$ (and, thus the number of features) is, in principle, infinite. However, given a training sample $I$, many of those possible cutoffs will yield exactly the same classification in the objects in $I$, which are the objects whose information is available. In this sense, for a certain predictor variable $\ell$ and a given training sample $I$, we can constrain the choice of $b \in \mathbb{R}$ to the finite set $B_\ell = \{x_\ell^u : u \in I\}$. In this way, the family of features under consideration is given by

$$\mathcal{F} = \{\phi_{\ell b} : b \in B_\ell, \ell = 1, 2, \ldots, p\}. \tag{2.3}$$

This family of features define the embedding

$$\Phi = (\phi_{\ell b})_{\left\{ \substack{b \in B_\ell \\ \ell = 1, 2, \ldots, p} \right\}}$$

used in the score function (1.5), which now becomes

$$f(x) = \omega^\top \Phi(x) + \beta = \sum_{\ell=1}^{p} \sum_{\{b \in B_\ell \mid x_\ell \geq b\}} \omega_{\ell b} + \beta, \tag{2.4}$$

where $\omega \in \mathbb{R}^{\sharp(\mathcal{F})}$.

Note that binary 0-1 variables can be accommodated to this framework easily, by taking $b = 1$. It might be less obvious for ordinal variables, i.e. qualitative variables whose values can be sorted according to some meaningful order $\succeq$ . For instance, a predictor variable $\ell$ taking the values {'big', 'medium', 'small'} yields the following queries of type (2.1): "Is $x_\ell \succeq$ big?", with affirmative answer for $x_\ell \in$ {'big'}; "Is $x_\ell \succeq$ medium?", with affirmative

answer for $x_\ell \in$ {'medium', 'small'}; and "Is $x_\ell \succeq$ small?" always affirmatively answered.

For nominal variables, where there exists no meaningful order for the $k$ values they take, queries of type (2.1) make no sense. In order to accommodate all types of variables to a common framework, a preprocessing step is needed where every nominal variable $\ell$ is replaced by $k$ new variables as follows: for every possible value $\hat{x}$ of the original nominal variable $\ell$, a new binary variable is built taking value one when $x_\ell$ is equal to $\hat{x}$ and zero otherwise. For instance, a variable $\ell$ taking values {'red','blue','green'} is replaced by three binary variables asking the questions 'is $x_\ell$ = red?', 'is $x_\ell$ = blue?' and 'is $x_\ell$ = green?'.

**Example 2.1:** In the Credit Screening Database, from the UCI Machine Learning Repository [6] (see Appendix A for further details), there are 15 variables, six of which are continuous (c), four binary (b) and five are nominal (n). Since no information about the meaning of the values is provided, we have considered that values cannot be sorted according to a meaningful order, and have encoded them as explained above. After the encoding, the original set of 15 variables, whose information about their names, types and the values they take, is given in Table 2.1, is replaced by a set of 43 variables.

| name | type | values |
|---|---|---|
| A1 | b | b, a |
| A2 | c | [13.75,76.75] |
| A3 | c | [0,28] |
| A4 | n | u, y, l, t |
| A5 | n | g, p, gg |
| A6 | n | c, d, cc, i, j, k, m, r, q, w, x, e, aa, ff |
| A7 | n | v, h, bb, j, n, z, dd, ff, o |
| A8 | c | [0,28.5] |
| A9 | b | t, f |
| A10 | b | t, f |
| A11 | c | [0,67] |
| A12 | b | t, f |
| A13 | n | g, p, s |
| A14 | c | [0,2000] |
| A15 | c | [0,100000] |

*Tab. 2.1:* Types of variables in Credit Screening Database

**Example 2.2:** In the Cylinder Bands Database, from the UCI Machine Learning Repository [6], there are 35 variables (excluding the first four attributes, which are for identification of the object), twenty of which are considered to be continuous (c), five binaries (b), three are considered to be ordinal (o) and eight are are considered to be nominal (n). All objects having at least one missing value have been removed from the database. After the encoding, the original set of 34 variables, whose information about its name, type and values it takes is given in Table 2.2, is replaced by a set of 56 variables.

For a certain predictor variable $\ell$, the coefficient $\omega_{\ell b}$ associated to feature $\phi_{\ell b}$ represents the amount with which the query 'is $x_\ell \geq b$?' contributes to the score function (2.4). Hence, the weight $\omega_{\ell b}$ gives insightful knowledge about how predictor variable $\ell$ influences the classification, since we are replacing variable $\ell$ by the function $s \mapsto \sum_{\{b \in B_\ell | \, s \geq b\}} \omega_{\ell b}$, thus determining the change of scale to be applied to variable $\ell$. Moreover, those variables $\ell$ for which $\omega_{\ell b}$ are zero for all $b \in B_\ell$ are not needed for the classification, and can be discarded.

**Example 2.3:** For instance, taking the Wisconsin Breast Cancer Database from the UCI Machine Learning Repository, [6], with data from cancer diagnosis (see Appendix A for further details of the database), and using the BSVM, it turns out that only 12 out of 30 variables have at least one non-zero $\omega_{\ell b}$. In other words, only 12 out of the 30 variables are relevant for the classification. In Figure 2.1 we show, for each of these twelve variables, its contribution to the score function. As an illustration, Table 2.3 shows, for variable `Worst Texture`, its cutoffs $b$, the corresponding weights $\omega_{\ell b}$ and the cumulative weights $\sum_{b' \leq b} \omega_{\ell b'}$.

We have also plotted in Figure 2.1 the median (represented by a star) and the mean (represented as a cross). It can be seen how, although the mean, or the median, is sometimes a good choice for the cutoff, this does not happen in general, and BSVM prefers other choices.

Since the output of the features proposed in this chapter is always binary, the importance, represented by the coefficients, is always measured in the same scale. The practitioner could choose to interpret those features with the highest coefficient in absolute value, obtaining in this way a great insight in the behavior of the classifier. However, if the practitioner prefers to keep

| id | name | type | values |
|---|---|---|---|
| 5 | grain screened | b | yes, no |
| 6 | ink color | b | key, type |
| 7 | proof on ctd ink | b | yes, no |
| 8 | blade mfg | n | benton, daetwyler, uddeholm |
| 9 | cylinder division | n | gallatin, warsaw, mattoon |
| 10 | paper type | n | uncoated, coated, super |
| 11 | ink type | n | uncoated, coated, cover |
| 12 | direct steam | b | yes, no |
| 13 | solvent type | n | xylol, lactol, naptha, line, other |
| 14 | type on cylinder | b | yes, no |
| 15 | press type | n | WoodHoe70, Motter70, Albert70, Motter94 |
| 16 | press | o | 802, 813, 815, 816, 821, 824, 827, 828 |
| 17 | unit number | o | 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 |
| 18 | cylinder size | n | catalog, spiegel, tabloid |
| 19 | paper mill location | n | northUS, southUS, canadian, scandanavian, mideuropean |
| 20 | plating tank | b | 1910, 1911, |
| 21 | proof cut | c | [0,100] |
| 22 | viscosity | c | [0,100] |
| 23 | caliper | c | [0,1.0] |
| 24 | ink temperature | c | [5,30 ] |
| 25 | humifity | c | [5,120] |
| 26 | roughness | c | [0,2] |
| 27 | blade pressure | c | [10,75] |
| 28 | varnish pct | c | [0,100] |
| 29 | press speed | c | [0,4000] |
| 30 | ink pct | c | [0,100] |
| 31 | solvent pct | c | [0,100] |
| 32 | ESA Voltage | c | [0,16] |
| 33 | ESA Amperage | c | [0,10] |
| 34 | wax | c | [0,4.0] |
| 35 | hardener | c | [0,3.0] |
| 36 | roller durometer | c | [15,120] |
| 37 | current density | c | [20,50] |
| 38 | anode space ratio | c | [70,130] |
| 39 | chrome content | c | [80,120] |

*Tab. 2.2:* Variables in Cylinder Bands Database

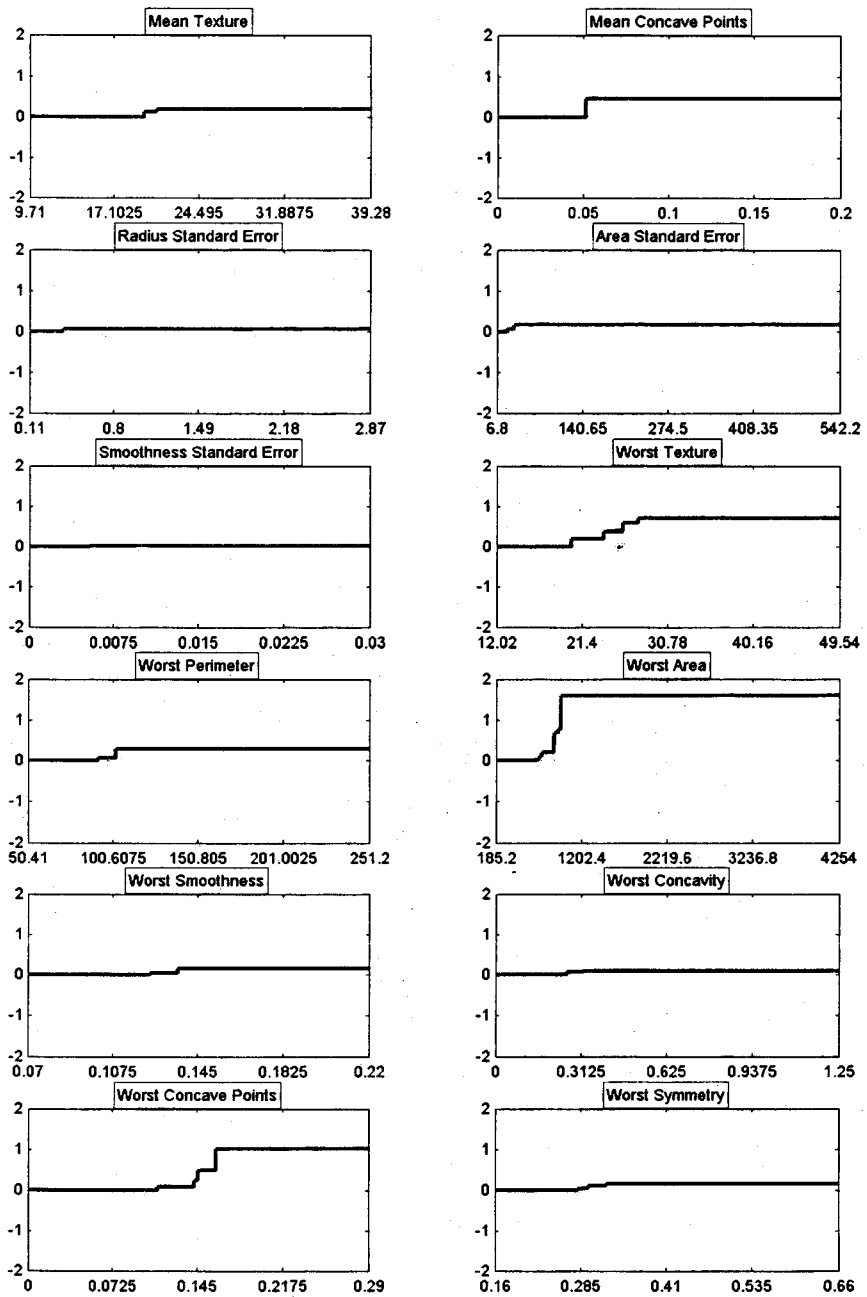


*Fig. 2.1:* Automatic choice of the scales

| $b$ | $\omega_{\ell b}$ | $\sum_{b'<b} \omega_{\ell b'}$ |
|---|---|---|
| 20.24 | 0.17795 | 0.17795 |
| 23.75 | 0.19729 | 0.37524 |
| 25.73 | 0.01160 | 0.38685 |
| 25.84 | 0.20116 | 0.58801 |
| 27.57 | 0.11219 | 0.70019 |

*Tab. 2.3:* Scale for the variable `Worst Texture`.

the number of features low in order to get a complete picture about how the classifier works, then a wrapper feature selection procedure, as in [35], could be used to gain interpretability. In Section 2.4.2 we give some numerical results using a simple but powerful wrapper procedure known as *recursive feature elimination*, as first proposed in [36].

We follow here the so-called *soft-margin* approach, described in Section 1.2.3, which allows some objects to be misclassified. This approach has been empirically shown to avoid overfitting, a phenomenon which happens when a low misclassification rate in $I$ does not generalize to forthcoming objects. Following Section 1.2.3, in order to obtain $\omega$ and $\beta$, we use the soft-margin maximization problem (1.11), which yields Linear Programming formulation (1.12).

After finding the maximal soft-margin hyperplane in the feature space defined by $\mathcal{F}$, the score function has the form described in (2.4).

For each feature, the absolute value of its coefficient indicates the importance of that feature for the classification. Using basic Linear Programming theory, it is easy to see that the number of features with non-zero coefficient is not larger than the number of objects in the database.

## 2.3 Building the classifier

In this section, we propose Problem (1.12) to be solved by the well-known Mathematical Programming tool called Column Generation, described in Section 1.4. It consists of solving a series of reduced problems, where decision variables are iteratively added as needed. Each decision variable $\omega$ corresponds to a feature $\phi$, so in each step of the column generation algorithm, the pricing problem chooses the most promising feature $\phi \in \mathcal{F}$.

For sake of completeness, we formulate now, the master problem (1.12-$F$)

for a given set of features $F$ :

$$\begin{array}{lrll} \min & \sum_{\phi\in F}(\omega_\phi^+ + \omega_\phi^-) + C\sum_{u\in I}\xi^u & & \\ \text{s.t.:} & \sum_{\phi\in F}(\omega_\phi^+ - \omega_\phi^-)c^u\phi(x^u) + \beta c^u + \xi^u \geq 1 & \forall u \in I & \\ & \omega_\phi^+ \geq 0 & \phi \in F & \\ & \omega_\phi^- \geq 0 & \phi \in F & \\ & \xi^u \geq 0 & \forall u \in I & \\ & \beta \in I\!R. & & \end{array} \qquad (1.12)\text{-}F$$

We start with an initial set of features $F$. For example, we can get as an initial set of features, one feature $\phi_{\ell b}$ per variable $\ell$, with $b$ equal to the median of variable $x_\ell$ in the objects of $I$. Then, Problem (1.12-$F$) is solved and the new step of the column generation algorithm, new promising features $\phi$ are sought in order to add them to $F$.

In order to generate new features, the Column Generation technique uses the dual formulation of Problem (1.12),

$$\begin{array}{lrll} \max & \sum_{u\in I}\lambda_u & & \\ \text{s.t.:} & -1 \leq \sum_{u\in I}\lambda_u c^u\phi(x^u) \leq 1 & \forall\phi \in \mathcal{F} & \\ & \sum_{u\in I}\lambda_u c^u = 0 & & \\ & 0 \leq \lambda_u \leq C & u \in I. & \end{array} \qquad (2.5)$$

The dual formulation of the Master Problem (1.12-$F$) only differs from this one in the first set of constraints, which should be attained for all features $\phi \in F$ instead of all features $\phi \in \mathcal{F}$.

Let $(\omega^*, \beta^*)$ be an optimal solution of Master Problem (1.12-$F$), and let $(\lambda_u^*)_{u\in I}$ be the values of the corresponding optimal dual solution. If the optimal solution of the Master Problem (1.12-$F$) is also optimal for Problem (1.12), then, for every feature $\phi \in \mathcal{F}$ the constraints of the Dual Problem (2.5) will hold, i.e.

$$-1 \leq \sum_{u\in I}\lambda_u^* c^u\phi(x^u) \leq 1.$$

Denote $\Gamma(\phi) = \sum_{u\in I}\lambda_u^* c^u\phi(x^u)$. If $(\omega^*, \beta^*)$ is not optimal for Problem (1.12), then the most violated constraint gives us information about which feature is promising and could be added to $F$, in the sense that adding such a feature to the set $F$ would yield, at that iteration, the highest improvement of the objective function. Thus, we wish to generate a new feature $\phi \in \mathcal{F}$ maximizing $|\Gamma(\phi)|$. Finding such a $\phi$ can be reduced to solving two optimization

problems:

$$\max_{\phi\in\mathcal{F}} \Gamma(\phi),$$
$$\min_{\phi\in\mathcal{F}} \Gamma(\phi).$$

In Section 2.3.1, a specific exact algorithm for solving these problems for the set of features $\mathcal{F}$, is developed. In Section 2.3.2, implementations details for the Column Generation Algorithm are given.

### 2.3.1 Generation of features

Finding the best $\phi \in \mathcal{F}$ is reduced to finding a predictor variable $\ell$ and a cutoff $b \in B_\ell$, such that $|\Gamma(\phi_{\ell b})|$, with $\phi_{\ell b}$ defined by (2.2), is maximal. In this section we describe an algorithm for, fixed the predictor variable $\ell$, finding the cutoff $b$ maximizing $\Gamma(\phi_{\ell b})$. Finding the minimum can be done in a similar way.

First, we sort all the objects in decreasing order by the values of the predictor variable $\ell$. Denote by $u(i)$ the object in $i$-th position. For simplicity, suppose there are not repeated values, i.e. $x_\ell^{u(1)} > x_\ell^{u(2)} > \ldots > x_\ell^{u(\sharp(I))}$. The case with repeated values will be analyzed later on.

Once $\ell$ is fixed and all the objects are sorted by the values of the predictor variable $\ell$, the value $\Gamma(\phi_{\ell b})$ can be efficiently calculated with a recursive procedure. Indeed, for certain $i \in \{1, 2, \ldots, \sharp(I)\}$, we have $\Gamma(\phi_{\ell b_{i+1}}) = \Gamma(\phi_{\ell b_i}) + \lambda^*_{u(i+1)} c^{u(i+1)}$, where $b_i$ denote the cutoff chosen as $x^{u(i)}$. Moreover, since $\lambda^*_u$ is nonnegative for all $u \in I$, whenever $c^u(i+1) = 1$, then $\Gamma(\phi_{\ell b_{i+1}}) > \Gamma(\phi_{\ell b_i})$. Thus, checking whether $\phi_{\ell b_i}$ is a maximum is not needed for every $i$, but only for those $i$ such that $c^{u(i)} = 1$ and $c^{u(i+1)} = -1$.

In the case in which there are repeated values in $\{x_\ell^u : u \in I\}$, the rule above does not apply. Let $i$ and $t$ be such that $x_\ell^{u(i-1)} > x_\ell^{u(i)} = x_\ell^{u(i+1)} = \ldots = x_\ell^{u(i+t)} > x_\ell^{u(i+t+1)}$. Note that, in the set of objects where predictor variable $\ell$ has the same value, there could be objects belonging to different classes. In this case, then $b = b_i$ must be checked, whatever the value of $c^{u(i+t+1)}$. However, if $c^{u(i)} = c^{u(i+1)} = \ldots = c^{u(i+t)}$ and $c^{u(i+t+1)} = 1$ we know that setting $b = b_j$, for any $j = i, i+1, \ldots, i+t$, will be improved by setting $b = b_{i+t+1}$. This means that $b = b_i$ does not give a maximum of $\Gamma(\phi_{\ell b})$. Only if $c^{u(i)} = 1$ and $c^{u(i+t+1)} = -1$, it is worth to consider $b = b_i$ as a candidate to be the maximum.

The minimization of $\Gamma$ is done analogously. For example, in case of no repeated values, candidates to be a minimum correspond to objects $u(i)$

belonging to class $-1$ where the next object $u(i+1)$ belongs to class 1.

Taking into account all these considerations we obtain, for a fixed predictor variable $\ell$, given the dual values $\lambda^*_u$, the algorithm described below, which finds the cutoff $b^+_\ell$ (and respectively, $b^-_\ell$) for which $\Gamma(\phi_{\ell b^+_\ell})$ (respectively, $\Gamma(\phi_{\ell b^-_\ell})$) is maximal (respectively, minimal).

**Algorithm 1-BSVM: Choosing a cutoff for variable $\ell$.**

**Step 0.** Sort the objects by $x_\ell : x_\ell^{u(i)}$.

**Step 1.** Set $i \leftarrow 1$, $sum \leftarrow 0$, $max \leftarrow 0$ and $min \leftarrow 0$.

**Step 2.** Set $sum \leftarrow sum + \lambda^*_{u(i)} c^{u(i)}$.

**Step 3. Step 3.1.** If $x_\ell^{u(i)} = x_\ell^{u(i+1)}$, then, go to Step 4.

**Step 3.2.** Otherwise, if for some $t > 0$, $x_\ell^{u(i-t-1)} < x_\ell^{u(i-t)} = \ldots = x_\ell^{u(i)} < x_\ell^{u(i)-1}$ and there exists $j$ with $j = 1, \ldots, t$ and $c^{u(i)} \neq c^{u(i-j)}$, then:

- If $sum > max$, then set $max \leftarrow sum$.
- If $sum < min$, then set $min \leftarrow sum$.

**Step 3.3.** Otherwise,

- if $c^{u(i)} = 1$, $c^{u(i+1)} = -1$ and $sum > max$, then set $max \leftarrow sum$.
- if $c^{u(i)} = -1$, $c^{u(i+1)} = 1$ and $sum < min$, then set $min \leftarrow sum$.

**Step 4.** Set $i \leftarrow i + 1$. If $i \leq \sharp(I)$, then go to Step 2, otherwise STOP.

### 2.3.2 *Implementation details*

The column generation algorithm has been implemented as follows. First, an initial set of features $F_0$ is built. We have chosen to start with one feature per variable, with the cutoff set equal to its median in the objects of $I$. Then, Problem (1.12-$F_0$) is solved for such initial set of features. The dual values of the optimal solution found, are used to generate new features.

In every step of the column generation algorithm, instead of generating just one feature (the one maximizing $|\Gamma(\phi)|$), we generate two features for every predictor variable $\ell$, given by the cutoffs for which $\Gamma(\phi_{\ell b})$ is maximal

and minimal. This is done using Algorithm 1, as described in Section 2.3.1. We do it for all the predictor variables, thus obtaining $2p$ features. Those generated features having $|\Gamma(\phi)| > 1$ are added to $F$ and the LP problem (1.12-$F$) is solved. These steps are repeated until all the generated features have $|\Gamma(\phi)| \leq 1$, in which case, we have found an optimal solution of Problem (1.12). A summary of this Column Generation Algorithm is described as follows.

**Algorithm CG-BSVM**

**Step 0.** Set $F_0 \leftarrow \{\phi_{1b_1^*}, \phi_{2b_2^*}, \ldots, \phi_{pb_p^*}\}$, where $b_\ell^*$ is the median of the predictor variable $\ell$, for $\ell = 1, 2, \ldots, p$. Set $F \leftarrow F_0$.

**Step 1.** Solve Problem (1.12-$F$). Let $(\omega^*, \beta^*)$ be its optimal solution, with dual values $\lambda_u^*$, $\forall u \in I$.

**Step 2.** For each $\ell = 1, 2, \ldots, p$ do:

**Step 2.1.** Run **Algorithm 1** to choose $b_\ell^+$.

**Step 2.2.** If $\Gamma(\phi_{\ell b_\ell^+}) > 1$, then set $F \leftarrow F \cup \{\phi_{\ell b_\ell^+}\}$.

**Step 2.3.** Run **Algorithm 1** to choose $b_\ell^-$.

**Step 2.4.** If $\Gamma(\phi_{\ell b_\ell^-}) < -1$, then set $F \leftarrow F \cup \{\phi_{\ell b_\ell^-}\}$.

**Step 3.** If $F$ has been modified, then go to Step 1, otherwise STOP: we have found an optimal solution of Problem (1.12).

## 2.4 Numerical results

First we are going to analyze the classification ability of the set of features proposed in the chapter. With this aim, a series of numerical experiments have been performed using databases publicly available from the UCI Machine Learning Repository [6]. Five different databases were used, namely, `bands`, `credit`, `ionosphere`, `sonar` and `wdbc`. Details about them are given in Appendix A.

In order to compare the quality of the BSVM classifier with the classification quality of other classifiers, we have tested the performance of two very different benchmark methods: Classification Trees, both with pruning

(prTree) and without pruning (Tree), and SVM with linear kernel. All results presented are obtained by 10-fold crossvalidation, e.g. [47]. The average percentages of correctly classified objects in both the training sample (`tr`) and testing sample (`test`) are displayed in Table 2.4 for different values of the parameter $C$. CPLEX 8.1.0 [42] was used as the LP solver.

It is a well-known fact in SVM that, if the parameter $C$ is chosen too close to zero, one may obtain as optimal solution of Problem (1.12) a vector with $\omega = 0$, from which a trivial classifier assigning all objects to one class is obtained. This degenerate situation (indicated in Tables 2.5-2.6 as `d.c.`) is avoided by taking a bigger $C$.

### 2.4.1 Comparing with other techniques

Results of the BSVM are shown in Table 2.5, where the average percentages of correctly classified objects in the training and testing samples are displayed along with the number of generated features (♯ `features`) with non-zero coefficient in the classifier, and the number of predictor variables actually used by the classifier (♯ `var`). As the results show, the BSVM behaves considerably better than Classification Trees and comparable to the standard linear SVM technique. Classification Trees are widely used in applied fields as diverse as medicine (diagnosis), computer science (data structures), botany (classification), and psychology (decision theory), mainly because they are easy to interpret. We claim that the BSVM maintains this property without losing the good classification ability of the standard linear SVM.

### 2.4.2 Reducing the number of features

The results in Table 2.5 show that the number of features is usually over one or even two hundreds, which makes hard to detect the most relevant features. In order to obtain a more interpretable classifier, we proceed with a pruning procedure in which features are recursively deleted. In this procedure, which has been successfully applied in standard SVM, see [36], all the generated features with zero coefficient in the classifier and the feature with non-zero coefficient having the smallest absolute value are eliminated. Then, the coefficients are recomputed by the optimization of the LP problem (1.12). This elimination procedure is repeated until the number of features is below a number given in advance.

The average percentages of correctly classified objects in the training and

| bands | | | |
|---|---|---|---|
| method | $C$ | % tr | % test |
| SVM | 0.01 | 64.44 | 64.44 |
| SVM | 0.10 | 74.57 | 65.93 |
| SVM | 1 | 80.16 | 71.85 |
| SVM | 10 | 81.65 | 72.96 |
| SVM | 100 | 82.18 | 72.22 |
| SVM | 1000 | 82.35 | 72.59 |
| prTree | | 74.12 | 64.81 |
| Tree | | 92.43 | 66.67 |
| **credit** | | | |
| method | $C$ | % tr | % test |
| SVM | 0.01 | 86.36 | 86.31 |
| SVM | 0.10 | 86.31 | 86.31 |
| SVM | 1 | 86.74 | 85.85 |
| SVM | 10 | 86.80 | 86.00 |
| SVM | 100 | 86.91 | 85.85 |
| SVM | 1000 | 87.09 | 85.54 |
| prTree | | 86.31 | 86.31 |
| Tree | | 95.15 | 81.69 |
| **ionosphere** | | | |
| method | $C$ | % tr | % test |
| SVM | 0.01 | 66.25 | 65.71 |
| SVM | 0.10 | 89.21 | 87.71 |
| SVM | 1 | 91.84 | 87.71 |
| SVM | 10 | 94.03 | 88.29 |
| SVM | 100 | 95.21 | 87.71 |
| SVM | 1000 | 95.65 | 86.29 |
| prTree | | 91.17 | 89.43 |
| Tree | | 98.03 | 87.71 |
| **sonar** | | | |
| method | $C$ | % tr | % test |
| SVM | 0.01 | 54.56 | 54.00 |
| SVM | 0.10 | 84.33 | 75.00 |
| SVM | 1 | 88.39 | 74.50 |
| SVM | 10 | 92.28 | 73.50 |
| SVM | 100 | 97.06 | 75.00 |
| SVM | 1000 | 100.00 | 73.50 |
| prTree | | 82.56 | 71.50 |
| Tree | | 97.56 | 77.00 |
| **wdbc** | | | |
| method | $C$ | % tr | % test |
| SVM | 0.01 | 87.16 | 86.79 |
| SVM | 0.10 | 95.95 | 95.71 |
| SVM | 1 | 98.27 | 98.04 |
| SVM | 10 | 98.45 | 97.68 |
| SVM | 100 | 99.11 | 96.96 |
| SVM | 1000 | 99.50 | 96.43 |
| prTree | | 96.71 | 94.46 |
| Tree | | 99.31 | 93.75 |

*Tab. 2.4:* Classification behavior in benchmark methods

| bands | | | | |
|---|---|---|---|---|
| $C$ | % tr | % test | ♯ features | ♯ var |
| 0.01 | d.c. | | | |
| 0.1 | d.c. | | | |
| 0.1 | 98.85 | 73.33 | 121.1 | 28.0 |
| 10 | 100.00 | 72.59 | 128.8 | 28.3 |
| 100 | 100.00 | 72.22 | 128.5 | 28.3 |
| 1000 | 100.00 | 72.59 | 128.4 | 28.4 |
| credit | | | | |
| $C$ | % tr | % test | ♯ features | ♯ var |
| 0.01 | 86.31 | 86.31 | 1.0 | 1.0 |
| 0.1 | 86.31 | 86.31 | 1.0 | 1.0 |
| 0.1 | 95.71 | 82.92 | 138.2 | 21.7 |
| 10 | 100.00 | 80.00 | 199.5 | 24.8 |
| 100 | 100.00 | 79.69 | 200.3 | 24.7 |
| 1000 | 100.00 | 80.31 | 200.5 | 24.8 |
| ionosphere | | | | |
| $C$ | % tr | % test | ♯ features | ♯ var |
| 0.01 | d.c. | | | |
| 0.1 | 91.17 | 90.57 | 2.0 | 2.0 |
| 0.1 | 100.00 | 90.57 | 92.7 | 31.1 |
| 10 | 100.00 | 90.57 | 93.1 | 31.2 |
| 100 | 100.00 | 90.57 | 93.1 | 31.2 |
| 1000 | 100.00 | 90.57 | 93.4 | 31.1 |
| sonar | | | | |
| $C$ | % tr | % test | ♯ features | ♯ var |
| 0.01 | d.c. | | | |
| 0.1 | 91.83 | 75.00 | 39.2 | 25.9 |
| 0.1 | 100.00 | 80.50 | 97.5 | 47.8 |
| 10 | 100.00 | 80.00 | 97.4 | 47.8 |
| 100 | 100.00 | 80.50 | 97.5 | 47.8 |
| 1000 | 100.00 | 80.50 | 97.5 | 47.8 |
| wdbc | | | | |
| $C$ | % tr | % test | ♯ features | ♯ var |
| 0.01 | 92.60 | 90.54 | 1.0 | 1.0 |
| 0.1 | 97.74 | 96.07 | 21.1 | 9.0 |
| 0.1 | 100.00 | 95.71 | 68.4 | 24.8 |
| 10 | 100.00 | 96.25 | 68.2 | 25.0 |
| 100 | 100.00 | 95.89 | 67.6 | 25.0 |
| 1000 | 100.00 | 96.07 | 68.0 | 25.0 |

*Tab. 2.5:* Classification behavior on BSVM

testing sample are shown, in Table 2.6, when the elimination procedure is applied until 30 or less features remain in the classification rule. As can be seen, the classification ability slightly deteriorates, but still keeps on being better than the one of the Classification Trees.

| BSVM (reducing number of features) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | bands | | credit | | ionosphere | | sonar | | wdbc | |
| $C$ | tr | test | tr | test | tr | test | tr | test | tr | test |
| 0.01 | d.c. | | 86.31 | 86.31 | d.c. | | d.c. | | 92.60 | 90.54 |
| 0.1 | d.c. | | 86.31 | 86.31 | 91.17 | 90.57 | 91.94 | 76.50 | 97.74 | 95.89 |
| 1 | 92.14 | 72.22 | 91.93 | 84.00 | 100.00 | 90.00 | 100.00 | 77.50 | 100.00 | 95.71 |
| 10 | 94.81 | 66.30 | 89.50 | 80.92 | 100.00 | 89.43 | 100.00 | 77.00 | 100.00 | 96.07 |
| 100 | 95.47 | 66.30 | 90.09 | 82.00 | 100.00 | 89.71 | 100.00 | 77.00 | 100.00 | 96.07 |
| 1000 | 95.14 | 66.67 | 91.42 | 80.77 | 100.00 | 89.43 | 100.00 | 77.00 | 100.00 | 96.25 |

*Tab. 2.6:* Reducing the number of features to at most 30

### 2.4.3 *Behavior in presence of outliers*

The classifier proposed in this chapter is based on threshold functions, thus it seems that extreme observations, with very high or very low values, will not have a strong influence in the classifier. To empirically test this fact, a series of experiments have been performed where some outliers were artificially introduced in the database. Every cell in the database `wdbc` was chosen to be an outlier with probability 0.05. Those cells chosen, were modified by adding $\rho$ times the range of its predictor variable, for $\rho = 10, 100, 1000$. In Tables 2.7 and 2.8 results of database `wdbc` are shown for the benchmark methods and for the BSVM. The classification ability of the linear SVM classifier dramatically worsens when introducing outliers, whereas the BSVM is hardly affected.

## 2.5 *Conclusions*

In this chapter, a new SVM-based tool for supervised classification has been proposed where the classifier gives insightful knowledge about the way the predictor variables influence the classification. Indeed, the nonlinearity behavior of the data is modeled by the BSVM classifier using simple queries,

| Benchmark methods with outliers. Database wdbc | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | original | | $\rho = 1$ | | $\rho = 10$ | | $\rho = 100$ | |
| method | $C$ | % tr | % test | % tr | % test | % tr | % test | % tr | % test |
| SVM | 0.01 | 87.16 | 86.79 | 65.26 | 65.18 | 63.21 | 63.21 | 63.21 | 63.21 |
| SVM | 0.10 | 95.95 | 95.71 | 90.24 | 88.75 | 64.13 | 63.21 | 63.47 | 62.86 |
| SVM | 1 | 98.27 | 98.04 | 91.53 | 90.18 | 65.65 | 59.64 | 64.94 | 62.86 |
| SVM | 10 | 98.45 | 97.68 | 92.24 | 88.57 | 64.54 | 57.68 | 64.96 | 61.61 |
| SVM | 100 | 99.11 | 96.96 | 92.42 | 88.57 | 64.72 | 57.50 | 64.94 | 60.89 |
| SVM | 1000 | 99.50 | 96.43 | 92.40 | 88.75 | 64.70 | 57.50 | 64.94 | 60.71 |
| prTree | | 96.71 | 94.46 | 96.31 | 93.04 | 95.71 | 92.14 | 95.99 | 92.32 |
| Tree | | 99.31 | 93.75 | 98.95 | 93.75 | 98.95 | 93.75 | 98.95 | 93.75 |

*Tab. 2.7:* Classification behavior of benchmark methods with outliers

| BSVM with outliers. Database wdbc | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | original | | $\rho = 1$ | | $\rho = 10$ | | $\rho = 100$ | |
| $C$ | % tr | % test | % tr | % test | % tr | % test | % tr | % test |
| 0.01 | 92.60 | 90.54 | 90.00 | 85.89 | 90.00 | 85.89 | 90.00 | 85.89 |
| 0.1 | 97.74 | 96.07 | 98.35 | 95.36 | 98.35 | 95.36 | 98.35 | 95.36 |
| 1 | 100.00 | 95.71 | 100.00 | 95.18 | 100.00 | 95.18 | 100.00 | 95.18 |
| 10 | 100.00 | 96.25 | 100.00 | 95.00 | 100.00 | 95.00 | 100.00 | 95.00 |
| 100 | 100.00 | 95.89 | 100.00 | 95.00 | 100.00 | 95.00 | 100.00 | 95.00 |
| 1000 | 100.00 | 96.07 | 100.00 | 95.18 | 100.00 | 95.18 | 100.00 | 95.18 |

*Tab. 2.8:* Classification behavior of BSVM with outliers

of type (2.1), easily interpretable by practitioners, for instance by representations similar to Figure 2.1. Its classification behavior, which is between SVM and Classification Trees, makes BSVM an interesting tool when a good classification ability is required, but interpretability of the results is an important issue. Our preliminary tests show that BSVM is much more robust than SVM against outliers.

The binarization procedure has been applied to each variable separately. If interactions between variables are expected to be relevant, more general binarization procedures might be considered. This issue is addressed in Chapter 3.

# 3. DETECTING RELEVANT INTERACTIONS

## *3.1 Introduction*

In Chapter 2, we have introduced the so-called Binarized Support Vector Machine (BSVM) method, where each variable is replaced by a set of binary variables obtained by queries of type (2.1). Although BSVM gives a powerful tool for detecting which are the most relevant variables, interactions between them are not taken into account. In this chapter, based on [17], we extend the BSVM in order to get classifiers that detect the interactions between variables which are useful to improve the classification performance. We call the proposed extension Non-linear Binarized Support Vector Machine (NBSVM).

To extend BSVM, in addition to considering queries of type (2.1), for all possible cutoffs $b$, NBSVM also considers the simultaneous positive answer to two or more of such queries, i.e.,

$$\boxed{\begin{array}{c}\text{Is predictor variable } \ell_1 \text{ greater than or equal to } b_1,\\ \text{and predictor variable } \ell_2 \text{ greater than or equal to } b_2,\\ \vdots\\ \text{and predictor variable } \ell_g \text{ greater than or equal to } b_g?\end{array}} \tag{3.1}$$

Special attention will be paid to the case $g = 2$, in which the pairwise interaction is explored by queries of type

$$\boxed{\begin{array}{c}\text{Is predictor variable } \ell_1 \text{ greater than or equal to } b_1,\\ \text{and predictor variable } \ell_2 \text{ greater than or equal to } b_2?\end{array}} \tag{3.2}$$

For this particular case, it is easy to measure how the classifier is affected by the interaction of each pair of variables, as shown in Section 3.2.

We restrict ourselves to the case in which two classes exist, $\mathcal{C} = \{-1, 1\}$. The multiclass case can be reduced to a series of two-class problems, as has been suggested e.g. in [39, 41, 68], as briefly described in Section 1.3.

The remainder of the chapter is organized as follows: NBSVM is described in Section 3.2. Since the number of features to be considered may be huge, the NBSVM method yields a large scale optimization problem with a large number of decision variables, in general, of order $(\sharp(I)p)^g$. The classifier uses the weighted sum of features defined from queries of type (3.1) and an independent term. In Section 3.3, the Column-Generation-based algorithm proposed in Chapter 2 is extended in order to solve the soft-margin maximization problem when queries of type (3.1) are considered. Numerical results are shown in Section 3.4, whereas conclusions are discussed in Section 3.5.

## 3.2 Nonlinear Binarized Support Vector Machines

The set of features $\mathcal{F}$ under consideration in BSVM, given by (2.3), does not consider interactions between predictor variables. Since we would like to get a model in which these interactions are taken into account, we now consider a family of features, $\hat{\mathcal{F}}$, obtained by multiplying different features of $\mathcal{F}$. We define $\hat{\mathcal{F}}$ as the set of all products of degree up to $g$ of features of $\mathcal{F}$, i.e.

$$\hat{\mathcal{F}} = \left\{ \prod_{\phi \in F} \phi : \; F \subset \mathcal{F}, \, \sharp(F) \leq g \right\}.$$

In what follows, features in $\mathcal{F}$ are called *features of degree one* whereas a feature $\phi \in \hat{\mathcal{F}}$, made up of the product of $k$ features of degree one, $\phi = \phi_{\ell_1,b_1} \cdot \phi_{\ell_2,b_2} \cdots \phi_{\ell_k,b_k}$ with $\ell_i \neq \ell_j \forall i \neq j$ and $b_\ell \neq \min_{u \in I} x^u_\ell$, is said to be a feature of degree $k$, for $k = 2, \ldots, g$.

In our model, each $\phi \in \hat{\mathcal{F}}$ is a feature that defines a component of the transformation $\Phi = (\phi)_{\phi \in \hat{\mathcal{F}}}$ used in the nonlinear score function (1.5).

Special attention will be paid to the case $g = 2$. For each pair of variables $(\ell_1, \ell_2)$ let $\phi_{\ell_1,\ell_2,b_1,b_2} = \phi_{\ell_1,b_1} \cdot \phi_{\ell_2,b_2}$. For simplicity in the notation, we denote $\omega_{\phi_{\ell_1,\ell_2,b_1,b_2}}$ and $\omega_{\phi_{\ell,b}}$ by $\omega_{\ell_1,\ell_2,b_1,b_2}$ and $\omega_{\ell,b}$, respectively. With this notation, the score function (2.4) can be rephrased as

$$f(x) = \sum_{\ell=1}^{p} \sum_{b \in B_\ell | x_\ell \geq b} \omega_{\ell b} + \sum_{\ell_1=1}^{p} \sum_{\ell_2=1}^{p} \sum_{b_2 \in B_{\ell_1} | x_{\ell_1} \geq b_1} \sum_{b_2 \in B_{\ell_2} | x_{\ell_2} \geq b_2} \omega_{\ell_1,\ell_2,b_1,b_2} + \beta. \tag{3.3}$$

The weight $\omega_{\ell b}$ associated to feature $\phi_{\ell b}$ represents the amount with which the query 'is $x_\ell \geq b$?' contributes to the score function (3.3). In Chapter 2, a

representation of how certain variable $\ell$ influences the classifier is proposed. We extend that representation to show the role that the interaction between a pair of variables plays in the classifier obtained with our procedure.

For a certain pair of variables $(\ell_1, \ell_2)$, the coefficient $\omega_{\ell_1,\ell_2,b_1,b_2}$ represents the amount with which the query 'is $x_{\ell_1} \geq b_1$ and simultaneously $x_{\ell_2} \geq b_2$?' contributes to the score function (3.3). The role played in the score function (3.3) by the interaction between variables $\ell_1$ and $\ell_2$, is represented by the function $\varphi_{\ell_1,\ell_2}(s,t)$,

$$\varphi_{\ell_1,\ell_2}(s,t) = \sum_{b_2 \in B_{\ell_1} | s \geq b_1} \sum_{b_2 \in B_{\ell_2} | t \geq b_2} \omega_{\ell_1,\ell_2,b_1,b_2}, \quad \forall (s,t) \in I\!R^2.$$

**Example 3.1:** Taking the Credit Screening Database, as in Example 2.1 (see Appendix A for further details of the database), and for $\omega$ obtained by the algorithm explained in Section 3.3, function $\varphi_{\texttt{A14,A3}}$, is plotted in Figure 3.1, where, at each point $(s,t)$ in the graphic, the gray intensity represents the value of $\varphi_{\texttt{A14,A3}}(s,t)$. The color in the down-left corner of the pictures corresponds to the null value, whereas lighter levels of gray corresponds to negative values and darker ones correspond to positive values.

The same information is given in Table 3.1. For instance, an object having $x^u_{\texttt{A3}} = 3$ and $x^u_{\texttt{A14}} = 800$, would have $\varphi_{\texttt{A14,A3}}(800,3) = \omega_{\texttt{A14,A3},210,0.375} = 0.6422$, represented by the darkest gray area in Figure 3.1, whereas an object having $x^u_{\texttt{A3}} = 15$ and $x^u_{\texttt{A14}} = 500$, would have $\varphi_{\texttt{A14,A3}}(400,15) = \omega_{\texttt{A14,A3},210,0.375} + \omega_{\texttt{A14,A3},232,5.835} + \omega_{\texttt{A14,A3},70,9.5} = 0.6422 - 0.2613 - 1.0642 = -0.6833$, represented by the light gray area in the top-right corner of Figure 3.1.

| values of A3 \ values of A14 | $[0,70)$ | $[70,210)$ | $[210,232)$ | $[232,2000]$ |
|---|---|---|---|---|
| $[0.000, 0.375)$ | 0 | 0 | 0 | 0 |
| $[0.375, 5.835)$ | 0 | 0 | 0.6422 | 0.6422 |
| $[5.835, 9.500)$ | 0 | 0 | 0.6422 | −0.4220 |
| $[9.500, 28.000)$ | 0 | −0.2613 | 0.3809 | −0.6833 |

*Tab. 3.1:* Role of the interaction in the score function

The interaction of those pair of variables $(\ell_1, \ell_2)$ having $\omega_{\ell_1,\ell_2,b_1,b_2} = 0$ for all $b_1 \in B_{\ell_1}$ and all $b_2 \in B_{\ell_2}$, has no effect in the classification based on score

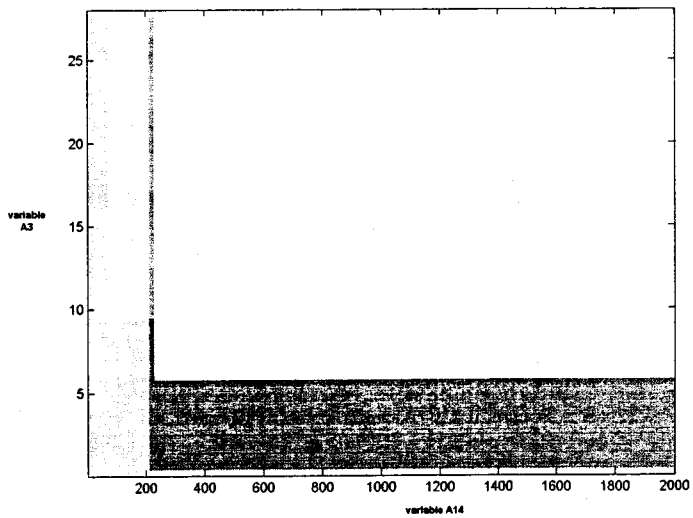


*Fig. 3.1:* Role of the interaction in the score function. Database `credit`

function (3.3). Moreover, the maximal absolute value that function $\varphi$ takes in $\mathbb{R}^2$ measures how important for the classification is the interaction of the pair of variables $(\ell_1, \ell_2)$. Such a measure, which we call *interaction intensity* is defined, for a pair of variables $(\ell_1, \ell_2)$ with $\ell_1 \neq \ell_2$, as follows

$$\begin{aligned} I(\ell_1, \ell_2) &= \max_{s,t \in \mathbb{R}} \left| \sum_{b_2 \in B_{\ell_1} | s \geq b_1} \sum_{b_2 \in B_{\ell_2} | t \geq b_2} \omega_{\ell_1, \ell_2, b_1, b_2} \right| \\ &= \max_{u \in \Omega} \left| \sum_{b_1 \in B_{\ell_1}} \sum_{b_2 \in B_{\ell_2}} \phi_{\ell_1, \ell_2, b_1, b_2}(x^u) \right| . \end{aligned}$$

Following a similar discussion, the importance for the classification of a variable $\ell$, without taking into account its interactions with other variables, is given by

$$I(\ell, \ell) = \max_{s \in \mathbb{R}} \left| \sum_{b \in B_\ell | s \geq b} \omega_{\ell, b} \right|$$

$$= \max_{u \in \Omega} \left| \sum_{b \in B_\ell} \phi_{\ell,b}(x^u) \right| .$$

**Example 3.2:** For the Credit Screening Database, Figure 3.2 represents in a gray scale the different values of $I(\ell_1, \ell_2)$ for each pair of variables $(\ell_1, \ell_2)$, for $\omega$ obtained by maximizing the soft margin, as will be explained in Section 3.3. Note that, nominal variables have been coded, as in Example 2.1. The identification number of the original variables, as described in Table 2.1, are shown at the edges of the table. From this picture, it is evident that in the classifier, the most relevant variable is the continuous variable `A8` and the pairs with highest interaction intensity $I$ are:

- nominal variable `A5` taking the value 'p', and continuous variable `A14`
- continuous variable `A2` and nominal variable `A7` taking the value 'v', and
- continuous variables `A2` and `A3`.

For many pairs of variables, interactions are discarded by the classifier: for instance, the interaction between continuous variables `A2` and `A11`, or nominal variables `A5` and `A6`.

**Example 3.3:** Taking the Cylinder Bands (see Appendix A for further details), whose variables are described in Example 2.2, Figure 3.4 represents, in a gray scale the different values of $I(\ell_1, \ell_2)$ for each pair of variables $(\ell_1, \ell_2)$, obtained after applying the method that will be explained in Section 3.3. For the nine pairs of variables $\ell_1$ and $\ell_2$ for which $I(\ell_1, \ell_2)$ is highest, function $\varphi$, is plotted in a gray scale in Figure 3.3. Black (respectively white) colors correspond to the highest (respectively lowest) value of $I(\ell_1, \ell_2)$ for any pair of variables.

For instance, looking at the graphic of variables `humifity` and `hardener` and going from the bottom-left corner in the top-right direction, the gray intensity becomes lighter and lighter. This occurs because features associated to those variables have negative weights. In the graphic of variables `ink pct` and `viscosity`, two weights are positive and one negative.

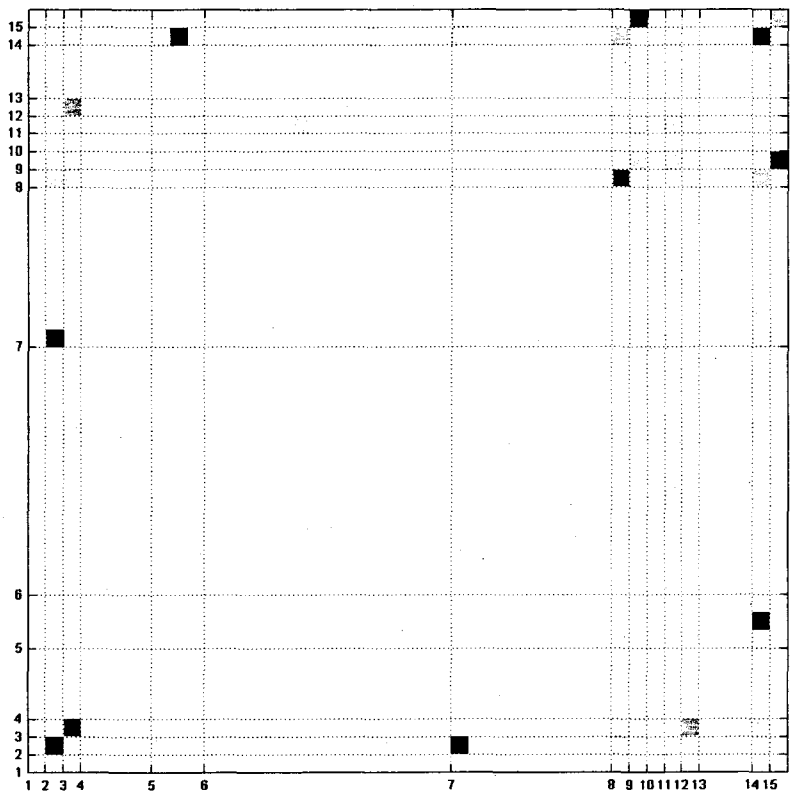


*Fig. 3.2:* Interaction between pairs of variables. Database `credit`

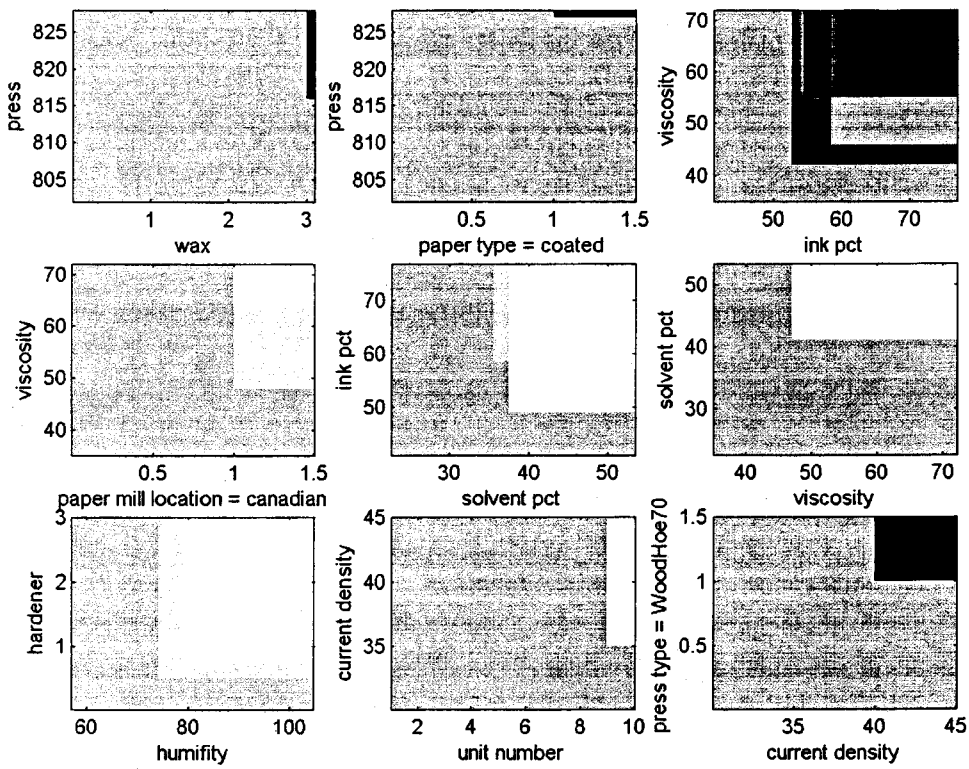


*Fig. 3.3:* Role of the interaction in the score function. Database bands

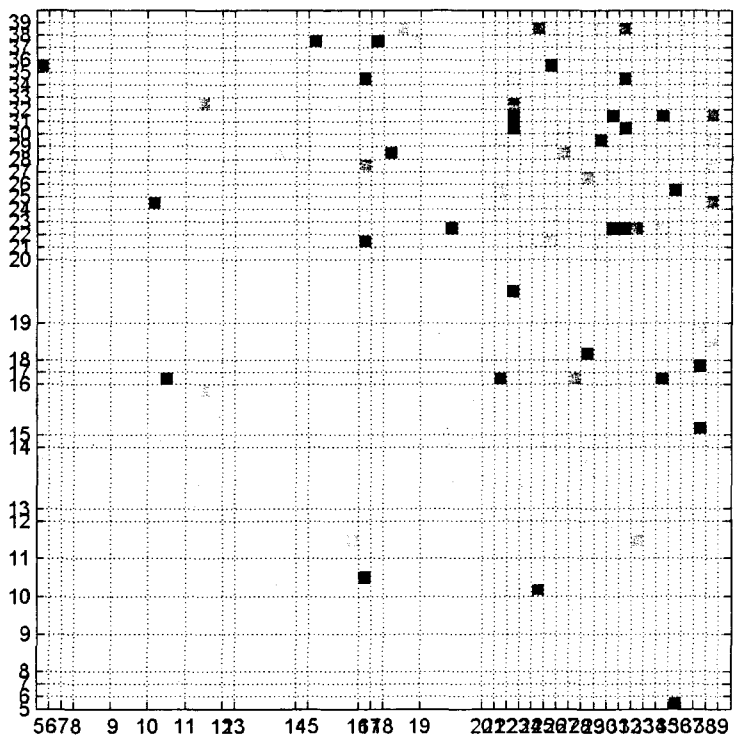

*Fig. 3.4:* Interaction between pairs of variables. Database bands

## 3.3 Building the classifier

In order to choose $\omega$ and $\beta$ in (2.4) we follow, as in Chapter 2, a soft-margin SVM-based approach [21], described in Section 1.2.3, which consists in finding the hyperplane which maximizes the margin in the feature space, but allowing some objects to be misclassified. As in the previous chapter, we choose the $L_1$ norm in the soft-margin maximization problem, yielding Problem (1.12-$\hat{\mathcal{F}}$) and use the column generation technique, described in Section 1.4 to solve it. After finding the maximal margin hyperplane in the feature space defined by $\hat{\mathcal{F}}$, the score function has the form described in (2.4).

In order to use, as in Chapter 2, the column generation technique, an algorithm for solving the pricing problem, which generates a promising feature in $\hat{\mathcal{F}}$, is needed. We develop such algorithm in Section 3.3.1 and, in Section 3.3.2, some implementation details for the column generation algorithm are given.

### 3.3.1 Generation of features

For the particular case in which just features of degree one are considered, an exact algorithm (Algorithm 1-BSVM, in Chapter 2) for solving the pricing problem is given. In such algorithm, for each predictor variable $\ell$, the most promising cutoff $b$ is found. This exact algorithm cannot be directly extended for features of degrees greater than one. For instance, when we want to generate a feature of degree 2, $\phi_{\ell_1,\ell_2,b_1,b_2}$, four parameters are to be determined: two predictor variables $\ell_1, \ell_2$, and two cutoffs $b_1, b_2$, one for each chosen predictor variable. If, as done in Algorithm 1-BSVM, we first consider the predictor variables $\ell_1, \ell_2$ are fixed, then it is not possible to sort the objects simultaneously according to two variables, so Algorithm 1-BSVM does not apply. Now we present a simple example where a local search heuristic procedure is used to generate promising features.

**Example 3.4:** In Table 3.2, a simple example with two predictor variables and four objects is presented, where we generate all the possible features. At a certain step of the column generation algorithm, let $\lambda_u$ be the dual values of the optimal solution of the master problem. We face the problem of generating the most promising feature. For predictor variable 1, the possible cutoffs are $5, 6$ and $9$ while the possible cutoffs for predictor variable 2 are $6, 8$ and $10$. We also consider here the trivial cutoff 3 for each predictor variable,

| class | $x_1$ | $x_2$ |
|---|---|---|
| $c^1$ | 5 | 8 |
| $c^2$ | 9 | 3 |
| $c^3$ | 6 | 6 |
| $c^4$ | 3 | 10 |

*Tab. 3.2:* Simple example of database

| | | | |
|---|---|---|---|
| $\lambda^1c^1+\lambda^2c^2+\lambda^3c^3+\lambda^4c^4$ | $\lambda^1c^1+\lambda^2c^2+\lambda^3c^3$ | $\lambda^2c^2+\lambda^3c^3$ | $\lambda^2c^2$ |
| $\lambda^1c^1+\lambda^3c^3+\lambda^4c^4$ | $\lambda^1c^1+\lambda^3c^3$ | $\lambda^3c^3$ | - |
| $\lambda^1c^1+\lambda^4c^4$ | $\lambda^1c^1$ | - | - |
| $\lambda^4c^4$ | - | - | - |

*Tab. 3.3:* Generated features of degrees one and two

which leads to a feature $\phi_{\ell,3}(x^u) = 1$ for all objects $u \in I$. In Table 3.3, the values of $\Gamma$ for different cutoffs for predictor variable 1 (columns) and different cutoffs for predictor variable 2 (rows) are shown.

Since we are also considering the trivial cutoffs, we can find in Table 3.3 the features of degree 1, represented in the first column and the first row of the table, along with the features of degree 2. Note that, when we move along the positions in the table, right-to-left and up-to-down movements lead to either $\Gamma$ gaining one term of the form $\lambda_u c^u$ or $\Gamma$ remaining itself. This will be taken into account in order to implement heuristics to find a promising feature $\phi$ to be added to $F$.

Suppose we know the value of $\Gamma$ for certain feature of degree two, $\phi = \phi_{\ell_1,\ell_2,b_1,b_2}$. In Example 3.4, this corresponds to a position in Table 3.3. Once a predictor variable is fixed, for instance the predictor variable $\ell_1$, the objects of $I$ can be sorted increasingly by their values in that predictor variable. Let $u(i)$ the object in the $i$-th position. We want to change the current cutoff $b_1 = x_{\ell_1}^{u(i)}$ in order to create a different feature $\hat{\phi} = \phi_{\ell_1,\ell_2,\hat{b}_1,b_2}$. When changing $b_1$ to a different value $\hat{b}_1 = x_{\ell}^{u(j)}$, the values of $\phi$ change only in the objects $u(k)$ with $i < k \leq j$ if $i < j$ (and, respectively $j < k \leq i$ if $j < i$). Taking this into account, we know that, when moving backward, i.e. $j < i$, then the

change in $\Gamma$ is easily computed using that

$$\Gamma(\hat{\phi}) = \Gamma(\phi) - \sum_{k:\,\phi(x^{u(k)})=1,\, j<k\leq i} \lambda^{u(k)} c^{u(k)}.$$

When moving forward, i.e. $j > i$, then, in order to know if the term $\lambda^{u(k)}c^{u(k)}$ must be added, we need to check, for all objects $u(k)$ with $i < k \leq j$, whether $x_{\ell_1}^{u(k)}$ is greater than or equal to the cutoff $\hat{b}_1$. When dealing with features of degree greater than two, this last condition should be checked for all the other variables used in the feature, but the rest remains analogous.

All the above comments are taken into account to implement a local search heuristic that, in every step, moves forward or backward in the ordered set of possible cutoffs for a randomly chosen variable. The algorithm stops when $T$ movements are performed without any improvement in the value of $\Gamma$.

**Algorithm 1-NBSVM: choosing $g$ cutoffs**

**Step 0.** Initialization:

- $i_k \leftarrow 1$, $\forall k = 1, 2, \ldots, g$.
- $b_k \leftarrow x_{\ell_k}^{u(i_k)}$, $\forall k = 1, 2, \ldots, g$.
- `steps` $\leftarrow 0$ and max$\leftarrow 0$.

**Step 1.** Randomly choose a predictor variable $\ell_j \in \{\ell_1, \ell_2, \ldots, \ell_g\}$.

**Step 2.** Randomly choose a type of movement: `forward` or `backward`.

**Step 3.**

- If `forward`, then randomly choose $h \in \{i_k, i_k + 1, \ldots, \sharp(I)\}$.
- If `backward`, then randomly choose $h \in \{1, 2, \ldots, i_k\}$.

**Step 4.** Let $b = x_{\ell_j}^{u(h)}$, compute $\Gamma(\phi)$, for $\phi = \phi_{\ell_1,\ldots,\ell_g,b_1,\ldots,b_{j-1},b,b_{j+1},\ldots,b_g}$.

**Step 5.** If $\Gamma(\phi) >$ `max` then $b^j \leftarrow b$, `max`$\leftarrow \Gamma(\phi)$ and `steps` $\leftarrow 0$. Otherwise, `steps` $\leftarrow$ `steps`+1.

**Step 6.** If `steps` <T, then go to Step 1. Otherwise, STOP.

An analogous algorithm for minimizing $\Gamma$ can be developed just changing Step 5 in the obvious way.

### 3.3.2 Implementation details

The column generation algorithm has been implemented as follows. First of all, the Algorithm CG-BSVM, described in Chapter 2, is run. Algorithm CG-BSVM obtains a solution of Problem (1.12) for the set $\mathcal{F}$ of features of degree one, and, as a byproduct, it also provides us with a set of features $F$, that have been generated during its application. We take such a set of features $F$, as an initial set $F \subset \mathcal{F} \subset \hat{\mathcal{F}}$ in Step 0 of the scheme presented in (CG-summary). In a second step, features of degree up to $g$ are generated. For a given set of $g$ variables $\{\ell_1, \ell_1, \ldots, \ell_g\}$, Algorithm 1-NBSVM described in Section 3.3.1 provides us with a local search heuristic to find the cutoffs $\{b_1, b_2, \ldots, b_g\}$, such that the feature $\phi = \phi_{\ell_1 b_1} \cdot \phi_{\ell_2 b_2} \cdot \ldots \cdot \phi_{\ell_g b_g}$ is promising to solve Problem (1.12).

In our implementation, $g$ predictor variables are randomly selected. Other ways of selecting the $g$ predictor variables might accelerate the optimization of Problem (1.12). We do not require the $g$ predictor variables to be different, allowing in this way features of degree lower than $g$. For instance if the feature generated is $\phi = \phi_{4,0.5} \cdot \phi_{7,7.3} \cdot \phi_{7,8.9}$, it can be simplified to $\phi = \phi_{4,0.5} \cdot \phi_{7,8.9}$ which is a feature of degree two. For such set of $g$ predictor variables, the local search heuristic described in Algorithm 1-NBSVM chooses their corresponding good cutoffs either by maximizing or minimizing $\Gamma$. We generate in this way $q$ features by maximizing $\Gamma$ with Algorithm 1-NBSVM for $q$ different random choices of the set of predictor variables, and other $q$ features by minimizing $\Gamma$ for other $q$ different random choices of the set of predictor variables. Among the generated features, those $\phi$ with $|\Gamma(\phi)| > 1$ are added to the set $F$ and the LP problem (1.12-$F$) is solved. The whole algorithm of column generation stops when all of these $2q$ generated features satisfy $|\Gamma(\phi)| < 1$. The current implementation of the column generation algorithm is as follows:

**Algorithm CG-NBSVM**

**Step 0.** Run Algorithm CG-BSVM to build an initial set $F$ with features of degree one.

**Step 1.** Repeat $q$ times:

**Step 1.1.** Randomly choose a set of $g$ predictor variables $\ell_1, \ell_2, \ldots, \ell_g$ and select $b_1, b_2, \ldots, b_g$ by maximizing $\Gamma$ using **Algorithm 1-NBSVM.**

**Step 1.2.** If $\Gamma(\phi) > 1$, then $F \leftarrow F \cup \{\phi\}$.

**Step 2.** Repeat $q$ times:

**Step 2.1.** Randomly choose a set of $g$ predictor variables $\ell_1, \ell_2, \ldots, \ell_g$ and use **Algorithm 1-NBSVM** for minimizing $\Gamma$, to choose $b_1, b_2, \ldots, b_g$.

**Step 2.2.** If $\Gamma(\phi) < -1$, then $F \leftarrow F \cup \{\phi\}$.

**Step 3.** If $F$ has not been modified in Steps 1 or 2, then STOP: we have found a good solution of Problem (1.12). Otherwise, solve Problem (1.12-$F$) and go to Step 1.

## 3.4 Numerical results

In this section we analyze the classification ability of the set of features proposed in this chapter. With this aim, a series of numerical experiments have been performed using the same databases used in Chapter 2, publicly available from the UCI Machine Learning Repository [6]. We use the five databases used in Chapter 2, namely, `bands`, `credit`, `ionosphere`, `sonar` and `wdbc` whose details are given in Appendix A.

In order to compare the quality of the NBSVM classifier with the classification quality of other classifiers, we have tested the performance of two very different benchmark methods: classification trees, with and without pruning, and SVM with linear kernel, polynomial kernel for degrees between two and five, and radial basis function kernel. The averaged percentages of correctly classified objects in both the training sample (`tr`) and testing sample (`test`) are displayed in Tables 3.5-3.9 for standard SVM and different values of the parameter $C$, which trades off the margin and the perturbations in formulation (1.9). For classification trees, results are shown in Table 3.4. All results presented are obtained by 10-fold crossvalidation, e.g. [47]. CPLEX 8.1.0 [42] was used as the LP solver.

| | bands | | credit | | ionosphere | | sonar | | wdbc | |
|---|---|---|---|---|---|---|---|---|---|---|
| | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test |
| Pruned Tree | 74.12 | 64.81 | 86.31 | 86.31 | 91.17 | 89.43 | 82.56 | 71.50 | 96.71 | 94.46 |
| Crude Tree | 92.43 | 66.67 | 95.15 | 81.69 | 98.03 | 87.71 | 97.56 | 77.00 | 99.31 | 93.75 |

*Tab. 3.4:* Results for Clasification Trees

| | lineal | | degree 2 | | degree 3 | | degree 4 | | degree 5 | | rbf | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $C$ | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test |
| 0.01 | 64.44 | 64.44 | 80.74 | 74.44 | 97.24 | 78.15 | 100.00 | 75.56 | 100.00 | 75.56 | 64.44 | 64.44 |
| 0.10 | 74.57 | 65.93 | 90.45 | 75.93 | 100.00 | 75.93 | 100.00 | 75.56 | 100.00 | 75.56 | 64.44 | 64.44 |
| 1 | 80.16 | 71.85 | 99.14 | 77.41 | 100.00 | 75.93 | 100.00 | 75.56 | 100.00 | 75.56 | 95.43 | 73.33 |
| 10 | 81.65 | 72.96 | 100.00 | 73.33 | 100.00 | 75.93 | 100.00 | 75.56 | 100.00 | 75.56 | 100.00 | 72.96 |
| 100 | 82.18 | 72.22 | 100.00 | 73.33 | 100.00 | 75.93 | 100.00 | 75.56 | 100.00 | 75.56 | 100.00 | 72.96 |
| 1000 | 82.35 | 72.59 | 100.00 | 73.33 | 100.00 | 75.93 | 100.00 | 75.56 | 100.00 | 75.56 | 100.00 | 72.96 |

*Tab. 3.5:* Results for SVM. Database bands

| | lineal | | degree 2 | | degree 3 | | degree 4 | | degree 5 | | rbf | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $C$ | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test |
| 0.01 | 86.36 | 86.31 | 86.70 | 86.15 | 92.99 | 85.54 | 96.44 | 84.92 | 98.77 | 81.69 | 54.77 | 54.77 |
| 0.10 | 86.31 | 86.31 | 90.87 | 85.08 | 96.36 | 84.00 | 98.79 | 80.46 | 99.62 | 77.69 | 74.53 | 70.77 |
| 1 | 86.74 | 85.85 | 95.16 | 84.92 | 98.24 | 80.46 | 99.52 | 77.85 | 99.71 | 77.38 | 95.08 | 85.69 |
| 10 | 86.80 | 86.00 | 96.96 | 83.85 | 99.32 | 77.69 | 99.71 | 76.77 | 99.78 | 77.38 | 97.42 | 85.85 |
| 100 | 86.91 | 85.85 | 98.44 | 78.62 | 99.73 | 77.08 | 99.78 | 76.92 | 100.00 | 76.15 | 99.13 | 83.85 |
| 1000 | 87.09 | 85.54 | 99.62 | 78.15 | 99.78 | 76.31 | 100.00 | 75.38 | 100.00 | 76.15 | 99.71 | 82.77 |

*Tab. 3.6:* Results for SVM. Database `credit`

| | lineal | | degree 2 | | degree 3 | | degree 4 | | degree 5 | | rbf | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $C$ | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test |
| 0.01 | 66.25 | 65.71 | 91.52 | 88.86 | 97.71 | 91.14 | 99.43 | 90.29 | 100.00 | 86.57 | 64.00 | 64.00 |
| 0.10 | 89.21 | 87.71 | 95.87 | 90.57 | 99.37 | 90.57 | 100.00 | 86.29 | 100.00 | 86.57 | 94.76 | 93.14 |
| 1 | 91.84 | 87.71 | 98.29 | 90.29 | 100.00 | 87.43 | 100.00 | 86.29 | 100.00 | 86.57 | 98.00 | 93.71 |
| 10 | 94.03 | 88.29 | 99.52 | 90.00 | 100.00 | 87.71 | 100.00 | 86.29 | 100.00 | 86.57 | 99.49 | 94.00 |
| 100 | 95.21 | 87.71 | 100.00 | 87.43 | 100.00 | 87.71 | 100.00 | 86.29 | 100.00 | 86.57 | 100.00 | 94.29 |
| 1000 | 95.65 | 86.29 | 100.00 | 87.43 | 100.00 | 87.71 | 100.00 | 86.29 | 100.00 | 86.57 | 100.00 | 94.29 |

*Tab. 3.7:* Results for SVM. Database `ionosphere`

| | lineal | | degree 2 | | degree 3 | | degree 4 | | degree 5 | | rbf | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $C$ | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test |
| 0.01 | 54.56 | 54.00 | 88.78 | 79.00 | 99.39 | 83.50 | 100.00 | 86.00 | 100.00 | 85.00 | 53.00 | 53.00 |
| 0.10 | 84.33 | 75.00 | 97.00 | 78.50 | 100.00 | 86.50 | 100.00 | 86.00 | 100.00 | 85.00 | 53.39 | 53.50 |
| 1 | 88.39 | 74.50 | 100.00 | 86.00 | 100.00 | 86.50 | 100.00 | 86.00 | 100.00 | 85.00 | 100.00 | 86.00 |
| 10 | 92.28 | 73.50 | 100.00 | 86.50 | 100.00 | 86.50 | 100.00 | 86.00 | 100.00 | 85.00 | 100.00 | 87.50 |
| 100 | 97.06 | 75.00 | 100.00 | 86.50 | 100.00 | 86.50 | 100.00 | 86.00 | 100.00 | 85.00 | 100.00 | 87.50 |
| 1000 | 100.00 | 73.50 | 100.00 | 86.50 | 100.00 | 86.50 | 100.00 | 86.00 | 100.00 | 85.00 | 100.00 | 87.50 |

*Tab. 3.8:* Results for SVM. Database `sonar`

| | lineal | | degree 2 | | degree 3 | | degree 4 | | degree 5 | | rbf | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $C$ | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test | % tr | % test |
| 0.01 | 87.16 | 86.79 | 95.36 | 95.18 | 96.37 | 96.25 | 98.06 | 97.86 | 98.31 | 97.68 | 76.35 | 76.25 |
| 0.10 | 95.95 | 95.71 | 97.80 | 97.14 | 98.25 | 98.21 | 98.65 | 97.50 | 99.19 | 96.96 | 95.54 | 95.36 |
| 1 | 98.27 | 98.04 | 98.39 | 97.68 | 98.97 | 97.68 | 99.44 | 96.79 | 99.92 | 96.61 | 98.35 | 97.86 |
| 10 | 98.45 | 97.68 | 99.25 | 97.86 | 99.66 | 96.96 | 100.00 | 96.43 | 100.00 | 96.07 | 99.13 | 98.21 |
| 100 | 99.11 | 96.96 | 99.98 | 96.43 | 100.00 | 96.25 | 100.00 | 96.43 | 100.00 | 96.07 | 100.00 | 96.96 |
| 1000 | 99.50 | 96.43 | 100.00 | 96.96 | 100.00 | 96.25 | 100.00 | 96.43 | 100.00 | 96.07 | 100.00 | 96.61 |

*Tab. 3.9:* Results for SVM. Database `wdbc`

| | | Crude | | | | Wrapped | |
|---|---|---|---|---|---|---|---|
| $g$ | $C$ | % tr | % test | ♯ f | ♯ v | % tr | % test |
| 1 | 0.01 | | d.c. | | | | d.c. |
| 1 | 0.1 | | d.c. | | | | d.c. |
| 1 | 1 | 98.85 | 73.33 | 121.1 | 28.0 | 92.14 | 72.22 |
| 1 | 10 | 100.00 | 72.59 | 128.8 | 28.3 | 94.81 | 66.30 |
| 1 | 100 | 100.00 | 72.22 | 128.5 | 28.3 | 95.47 | 66.30 |
| 1 | 1000 | 100.00 | 72.59 | 128.4 | 28.4 | 95.14 | 66.67 |
| 2 | 0.01 | | d.c. | | | | d.c. |
| 2 | 0.1 | | d.c. | | | | d.c. |
| 2 | 1 | 100.00 | 72.59 | 132.1 | 33.8 | 99.34 | 72.22 |
| 2 | 10 | 100.00 | 77.78 | 131.5 | 34.9 | 100.00 | 72.22 |
| 2 | 100 | 100.00 | 75.93 | 133.2 | 34.7 | 100.00 | 73.33 |
| 2 | 1000 | 100.00 | 75.56 | 134.2 | 34.6 | 100.00 | 75.56 |
| 3 | 0.01 | | d.c. | | | | d.c. |
| 3 | 0.1 | 88.23 | 73.33 | 39.0 | 28.0 | 87.41 | 72.96 |
| 3 | 1 | 100.00 | 74.07 | 140.9 | 35.6 | 99.92 | 76.30 |
| 3 | 10 | 100.00 | 76.30 | 143.4 | 35.0 | 100.00 | 70.74 |
| 3 | 100 | 100.00 | 77.41 | 143.2 | 34.8 | 100.00 | 74.07 |
| 3 | 1000 | 100.00 | 78.15 | 141.6 | 35.6 | 100.00 | 72.96 |
| 4 | 0.01 | | d.c. | | | | d.c. |
| 4 | 0.1 | 89.30 | 72.59 | 43.6 | 29.7 | 89.14 | 71.85 |
| 4 | 1 | 100.00 | 76.30 | 142.4 | 35.8 | 99.84 | 71.48 |
| 4 | 10 | 100.00 | 76.30 | 148.4 | 35.3 | 100.00 | 74.81 |
| 4 | 100 | 100.00 | 72.59 | 145.0 | 35.3 | 100.00 | 71.48 |
| 4 | 1000 | 100.00 | 78.52 | 147.4 | 35.0 | 100.00 | 73.33 |
| 5 | 0.01 | | d.c. | | | | d.c. |
| 5 | 0.1 | 90.33 | 71.11 | 55.7 | 31.0 | 89.67 | 71.11 |
| 5 | 1 | 100.00 | 77.04 | 148.0 | 35.2 | 100.00 | 75.19 |
| 5 | 10 | 100.00 | 75.56 | 146.0 | 35.2 | 100.00 | 75.19 |
| 5 | 100 | 100.00 | 77.78 | 144.3 | 35.0 | 100.00 | 74.07 |
| 5 | 1000 | 100.00 | 77.78 | 148.2 | 35.3 | 100.00 | 73.33 |

*Tab. 3.10:* Classification behavior. Database **bands**

Results of NBSVM are shown, for different values of $C$ and $g$, in Tables 3.10-3.14, where the averaged percentages of correctly classified objects in training and testing samples are displayed along with the number of generated features (♯ f) with non-zero coefficient in the classifier and the number of predictor variables actually used by the classifier (♯ v). As NBSVM is an extension of BSVM, we have included the results of BSVM, which coincides with NBSVM for $g = 1$, in the first group of rows in Tables 3.10-3.14. We have chosen parameters $T$ and $q$ in Algorithms 1-NBSVM and CL-NBSVM

| | | Crude | | | | Wrapped | |
|---|---|---|---|---|---|---|---|
| $g$ | $C$ | % tr | % test | ♯ f | ♯ v | % tr | % test |
| 1 | 0.01 | 86.31 | 86.31 | 1.0 | 1.0 | 86.31 | 86.31 |
| 1 | 0.1 | 86.31 | 86.31 | 1.0 | 1.0 | 86.31 | 86.31 |
| 1 | 1 | 95.71 | 82.92 | 138.2 | 21.7 | 91.93 | 84.00 |
| 1 | 10 | 100.00 | 80.00 | 199.5 | 24.8 | 89.50 | 80.92 |
| 1 | 100 | 100.00 | 79.69 | 200.3 | 24.7 | 90.09 | 82.00 |
| 1 | 1000 | 100.00 | 80.31 | 200.5 | 24.8 | 91.42 | 80.77 |
| 2 | 0.01 | 86.31 | 86.31 | 1.0 | 1.0 | 86.31 | 86.31 |
| 2 | 0.1 | 86.31 | 86.31 | 1.0 | 1.0 | 86.31 | 86.31 |
| 2 | 1 | 99.56 | 83.85 | 169.2 | 30.7 | 95.35 | 84.15 |
| 2 | 10 | 100.00 | 82.92 | 180.1 | 31.2 | 95.06 | 83.38 |
| 2 | 100 | 100.00 | 82.15 | 182.7 | 30.8 | 94.97 | 83.85 |
| 2 | 1000 | 100.00 | 81.69 | 181.8 | 30.3 | 95.11 | 81.23 |
| 3 | 0.01 | 86.31 | 86.31 | 1.0 | 1.0 | 86.31 | 86.31 |
| 3 | 0.1 | 86.32 | 86.00 | 1.0 | 1.1 | 86.32 | 86.00 |
| 3 | 1 | 99.93 | 83.54 | 188.4 | 29.0 | 95.86 | 83.54 |
| 3 | 10 | 100.00 | 85.85 | 193.7 | 29.3 | 96.48 | 81.85 |
| 3 | 100 | 100.00 | 83.85 | 192.6 | 29.5 | 96.92 | 82.31 |
| 3 | 1000 | 100.00 | 84.15 | 190.3 | 29.1 | 96.44 | 80.92 |
| 4 | 0.01 | 86.31 | 86.31 | 1.0 | 1.0 | 86.31 | 86.31 |
| 4 | 0.1 | 86.32 | 86.00 | 1.0 | 1.1 | 86.32 | 86.00 |
| 4 | 1 | 99.97 | 84.31 | 204.2 | 29.3 | 96.50 | 82.92 |
| 4 | 10 | 100.00 | 84.31 | 206.1 | 29.4 | 97.01 | 81.23 |
| 4 | 100 | 100.00 | 85.69 | 199.9 | 28.6 | 97.38 | 81.69 |
| 4 | 1000 | 100.00 | 85.54 | 198.5 | 28.5 | 97.74 | 81.85 |
| 5 | 0.01 | 86.31 | 86.31 | 1.0 | 1.0 | 86.31 | 86.31 |
| 5 | 0.1 | 86.32 | 86.00 | 1.0 | 1.1 | 86.32 | 86.00 |
| 5 | 1 | 100.00 | 84.92 | 209.1 | 28.4 | 96.27 | 83.38 |
| 5 | 10 | 100.00 | 85.38 | 213.0 | 27.8 | 97.44 | 80.46 |
| 5 | 100 | 100.00 | 83.85 | 207.8 | 28.3 | 97.45 | 80.46 |
| 5 | 1000 | 100.00 | 85.08 | 209.5 | 28.2 | 98.10 | 82.15 |

*Tab. 3.11:* Classification behavior. Database `credit`

| | | Crude | | | | Wrapped | |
|---|---|---|---|---|---|---|---|
| $g$ | $C$ | % tr | % test | ♯ f | ♯ v | % tr | % test |
| 1 | 0.01 | | d.c. | | | | d.c. |
| 1 | 0.1 | 91.17 | 90.57 | 2.0 | 2.0 | 91.17 | 90.57 |
| 1 | 0.1 | 100.00 | 90.57 | 92.7 | 31.1 | 100.00 | 90.00 |
| 1 | 10 | 100.00 | 90.57 | 93.1 | 31.2 | 100.00 | 89.43 |
| 1 | 100 | 100.00 | 90.57 | 93.1 | 31.2 | 100.00 | 89.71 |
| 1 | 1000 | 100.00 | 90.57 | 93.4 | 31.1 | 100.00 | 89.43 |
| 2 | 0.01 | | d.c. | | | | d.c. |
| 2 | 0.1 | 94.10 | 90.57 | 11.2 | 10.1 | 94.10 | 90.57 |
| 2 | 0.1 | 100.00 | 92.57 | 105.6 | 32.4 | 100.00 | 91.14 |
| 2 | 10 | 100.00 | 92.57 | 107.3 | 31.9 | 100.00 | 91.71 |
| 2 | 100 | 100.00 | 92.57 | 109.5 | 32.0 | 100.00 | 91.71 |
| 2 | 1000 | 100.00 | 92.00 | 108.1 | 32.1 | 100.00 | 91.71 |
| 3 | 0.01 | | d.c. | | | | d.c. |
| 3 | 0.1 | 94.89 | 90.86 | 16.5 | 13.3 | 94.89 | 90.57 |
| 3 | 0.1 | 100.00 | 92.29 | 101.2 | 32.6 | 100.00 | 91.43 |
| 3 | 10 | 100.00 | 92.29 | 104.4 | 32.7 | 100.00 | 92.29 |
| 3 | 100 | 100.00 | 92.86 | 103.0 | 32.8 | 100.00 | 91.71 |
| 3 | 1000 | 100.00 | 92.00 | 104.5 | 32.6 | 100.00 | 90.29 |
| 4 | 0.01 | | d.c. | | | | d.c. |
| 4 | 0.1 | 95.05 | 91.71 | 20.2 | 16.2 | 95.05 | 92.29 |
| 4 | 0.1 | 100.00 | 92.57 | 104.1 | 33.0 | 100.00 | 92.57 |
| 4 | 10 | 100.00 | 92.00 | 99.5 | 33.0 | 100.00 | 91.43 |
| 4 | 100 | 100.00 | 92.57 | 102.5 | 33.0 | 100.00 | 92.57 |
| 4 | 1000 | 100.00 | 92.29 | 102.4 | 33.0 | 100.00 | 93.14 |
| 5 | 0.01 | | d.c. | | | | d.c. |
| 5 | 0.1 | 95.59 | 92.57 | 35.1 | 24.8 | 95.62 | 92.57 |
| 5 | 0.1 | 100.00 | 93.43 | 100.8 | 32.8 | 100.00 | 91.71 |
| 5 | 10 | 100.00 | 92.29 | 101.5 | 33.0 | 100.00 | 92.57 |
| 5 | 100 | 100.00 | 93.14 | 98.7 | 33.0 | 100.00 | 92.57 |
| 5 | 1000 | 100.00 | 93.43 | 101.9 | 33.0 | 100.00 | 93.14 |

*Tab. 3.12:* Classification behavior. Database `ionosphere`

| | | Crude | | | | Wrapped | |
|---|---|---|---|---|---|---|---|
| $g$ | $C$ | % tr | % test | ♯ f | ♯ v | % tr | % test |
| 1 | 0.01 | | d.c. | | | d.c. | |
| 1 | 0.1 | 91.83 | 75.00 | 39.2 | 25.9 | 91.94 | 76.50 |
| 1 | 0.1 | 100.00 | 80.50 | 97.5 | 47.8 | 100.00 | 77.50 |
| 1 | 10 | 100.00 | 80.00 | 97.4 | 47.8 | 100.00 | 77.00 |
| 1 | 100 | 100.00 | 80.50 | 97.5 | 47.8 | 100.00 | 77.00 |
| 1 | 1000 | 100.00 | 80.50 | 97.5 | 47.8 | 100.00 | 77.00 |
| 2 | 0.01 | | d.c. | | | d.c. | |
| 2 | 0.1 | 99.06 | 80.50 | 63.7 | 48.4 | 98.44 | 81.50 |
| 2 | 0.1 | 100.00 | 85.00 | 109.0 | 55.6 | 100.00 | 84.00 |
| 2 | 10 | 100.00 | 85.00 | 107.7 | 56.7 | 100.00 | 84.00 |
| 2 | 100 | 100.00 | 86.00 | 107.6 | 56.3 | 100.00 | 82.50 |
| 2 | 1000 | 100.00 | 86.00 | 107.6 | 56.3 | 100.00 | 82.50 |
| 3 | 0.01 | | d.c. | | | d.c. | |
| 3 | 0.1 | 99.50 | 81.00 | 78.5 | 55.5 | 99.22 | 79.00 |
| 3 | 0.1 | 100.00 | 87.00 | 116.6 | 59.0 | 100.00 | 83.50 |
| 3 | 10 | 100.00 | 83.00 | 114.7 | 59.5 | 100.00 | 79.00 |
| 3 | 100 | 100.00 | 84.50 | 113.1 | 59.4 | 100.00 | 83.50 |
| 3 | 1000 | 100.00 | 84.50 | 113.1 | 59.4 | 100.00 | 83.50 |
| 4 | 0.01 | | d.c. | | | d.c. | |
| 4 | 0.1 | 99.83 | 82.00 | 97.0 | 59.1 | 99.61 | 78.50 |
| 4 | 0.1 | 100.00 | 84.00 | 119.7 | 59.8 | 100.00 | 81.50 |
| 4 | 10 | 100.00 | 83.00 | 119.0 | 59.7 | 100.00 | 80.00 |
| 4 | 100 | 100.00 | 83.00 | 118.9 | 59.8 | 100.00 | 80.00 |
| 4 | 1000 | 100.00 | 83.00 | 118.9 | 59.8 | 100.00 | 80.00 |
| 5 | 0.01 | | d.c. | | | d.c. | |
| 5 | 0.1 | 99.67 | 80.50 | 100.4 | 59.9 | 99.61 | 80.00 |
| 5 | 0.1 | 100.00 | 82.00 | 122.9 | 60.0 | 100.00 | 79.50 |
| 5 | 10 | 100.00 | 84.00 | 123.9 | 59.8 | 100.00 | 79.00 |
| 5 | 100 | 100.00 | 79.00 | 120.0 | 59.7 | 100.00 | 77.00 |
| 5 | 1000 | 100.00 | 79.00 | 120.0 | 59.7 | 100.00 | 77.00 |

*Tab. 3.13:* Classification behavior. Database `sonar`

| | | Crude | | | | Wrapped | |
|---|---|---|---|---|---|---|---|
| $g$ | $C$ | % tr | % test | ♯ f | ♯ v | % tr | % test |
| 1 | 0.01 | 92.60 | 90.54 | 1.0 | 1.0 | 92.60 | 90.54 |
| 1 | 0.1 | 97.74 | 96.07 | 21.1 | 9.0 | 97.74 | 95.89 |
| 1 | 0.1 | 100.00 | 95.71 | 68.4 | 24.8 | 100.00 | 95.71 |
| 1 | 10 | 100.00 | 96.25 | 68.2 | 25.0 | 100.00 | 96.07 |
| 1 | 100 | 100.00 | 95.89 | 67.6 | 25.0 | 100.00 | 96.07 |
| 1 | 1000 | 100.00 | 96.07 | 68.0 | 25.0 | 100.00 | 96.25 |
| 2 | 0.01 | 93.49 | 91.07 | 1.0 | 1.8 | 93.49 | 91.07 |
| 2 | 0.1 | 98.95 | 95.89 | 41.0 | 23.3 | 99.01 | 95.89 |
| 2 | 0.1 | 100.00 | 96.79 | 87.4 | 29.7 | 100.00 | 96.79 |
| 2 | 10 | 100.00 | 97.14 | 90.0 | 29.4 | 100.00 | 97.14 |
| 2 | 100 | 100.00 | 97.32 | 88.0 | 29.5 | 100.00 | 96.79 |
| 2 | 1000 | 100.00 | 96.61 | 89.9 | 29.5 | 100.00 | 96.61 |
| 3 | 0.01 | 93.67 | 91.07 | 1.0 | 2.6 | 93.67 | 91.07 |
| 3 | 0.1 | 98.95 | 95.00 | 42.6 | 25.5 | 98.95 | 95.36 |
| 3 | 0.1 | 100.00 | 96.61 | 103.0 | 29.7 | 100.00 | 95.89 |
| 3 | 10 | 100.00 | 97.14 | 99.9 | 30.0 | 100.00 | 96.25 |
| 3 | 100 | 100.00 | 97.50 | 103.3 | 30.0 | 100.00 | 96.61 |
| 3 | 1000 | 100.00 | 96.79 | 101.5 | 29.9 | 100.00 | 96.25 |
| 4 | 0.01 | 94.15 | 93.04 | 1.0 | 3.4 | 94.15 | 93.04 |
| 4 | 0.1 | 99.19 | 95.36 | 54.8 | 29.5 | 99.21 | 94.82 |
| 4 | 0.1 | 100.00 | 97.14 | 110.1 | 30.0 | 100.00 | 96.61 |
| 4 | 10 | 100.00 | 96.07 | 112.3 | 30.0 | 100.00 | 95.54 |
| 4 | 100 | 100.00 | 97.14 | 108.2 | 30.0 | 100.00 | 96.61 |
| 4 | 1000 | 100.00 | 97.50 | 112.1 | 30.0 | 100.00 | 96.61 |
| 5 | 0.01 | 94.15 | 91.79 | 1.0 | 4.0 | 94.15 | 91.79 |
| 5 | 0.1 | 99.23 | 95.54 | 54.2 | 27.8 | 99.19 | 95.89 |
| 5 | 0.1 | 100.00 | 95.89 | 114.4 | 30.0 | 100.00 | 95.18 |
| 5 | 10 | 100.00 | 96.25 | 110.8 | 30.0 | 100.00 | 96.43 |
| 5 | 100 | 100.00 | 96.43 | 113.3 | 30.0 | 100.00 | 96.07 |
| 5 | 1000 | 100.00 | 96.61 | 115.1 | 30.0 | 100.00 | 95.89 |

*Tab. 3.14:* Classification behavior. Database `wdbc`

as 100 and 25 respectively.

In order to interpret the obtained classifier, the number of features cannot be high. It might be useful to keep low the number of features actually used by the classifier. As in Section 2.4.2, it is also possible to reduce the number of features by using a wrapper approach that recursively deletes features. For instance, [36] proposes a procedure, successfully applied in standard linear SVM, where all the generated features with zero coefficient in the classifier and the feature whose non-zero coefficient has smallest absolute value are eliminated. Then, the coefficients are recomputed by the optimization of the LP Problem (1.12). This elimination procedure is repeated until the number of features is below a desired threshold. In Tables 3.10-3.14, numerical results are shown (columns `wrapped`), for an elimination procedure that is applied until 30 or less features remain in the classification rule.

It is a well-known fact in SVM that, if the parameter $C$ is chosen too close to zero, one may obtain as optimal solution of Problem (1.12) a vector with $\omega = 0$, from which a trivial classifier assigning all objects to one class is obtained. This degenerate situation (indicated in Tables 3.10-3.14 as `d.c.`) is avoided by taking a bigger $C$.

The results show that NBSVM behaves comparable to the standard SVM technique. Indeed the best averaged percentage of correctly classified objects, for the best choice of $C$, of standard SVM is never worse than NBSVM's more than 0.86%.

Comparing NBSVM with its ancestor BSVM, we see that the consideration of interactions via the introduction of features of degree greater than one leads to an improvement in the classification performance, together with the added value of allowing us to measure interactions intensity, as illustrated in Section 3.2.

Moreover, comparing Table 3.4 with Tables 3.10-3.14, it turns out that, in terms of classification performance, NBSVM generally behaves considerably better than classification trees. For instance, taking the database `bands`, for any choice of $C$ the prediction rate of NBSVM is nearly 5 points higher than Classification Trees.

If we want to keep low the number of features used, we can use the wrapped (instead of the crude) version of NBSVM. From our computational experience it seems that the wrapping slightly worsens the classification ability in most instances, though in some cases it deteriorates significantly. This is the case, for instance, of `sonar`. However, even the wrapped version behaves better or equal than Classification Trees (even 7 points above) for all

values of the parameter $C$, as shown in Tables 3.4 and 3.13.

## 3.5 Conclusions

In this chapter an extension of BSVM for supervised classification, the NBSVM, has been proposed, where the classifier gives insightful knowledge about the way the predictor variables and the interactions between them influence the classification. Indeed, the nonlinearity behavior of the data and interactions between the different variables are modeled by the NBSVM classifier using simple queries, of type (3.1), and combinations of them, easily interpretable by practitioners, for instance by representations similar to Figure 3.1. Its classification behavior, which is between SVM and Classification Trees, makes NBSVM an interesting tool when a good classification ability is required, but interpretability of the results is an important issue.

Although in most instances the wrapping procedure leads to a very slight deterioration of the classification ability, in some cases the change is considerable. The design of more sophisticated wrapping strategies deserves further study.

# 4. BIOBJECTIVE MARGIN MAXIMIZATION

## *4.1 Introduction*

Crude SVM, as described in Chapter 1, cannot take into account different misclassification costs or known a priori probabilities. In this chapter, based on our reference [14], we formulate a new model in which margin of a hyperplane on each class is dealt independently of the other class. We study the two-class problem $\mathcal{C} = \{-1, 1\}$ in which the simultaneous maximization of both margins is sought. In other words, we seek the set of hyperplanes such that there is not any other hyperplane having greater margin for both classes, thus we expect their performance cannot be improved simultaneously with respect to both classes.

As in Chapter 1, we deal separately with the hard-margin approach (Section 4.2), where separability is required, and the soft margin approach (Section 4.3), which allows nonseparability of $I$ and avoids overfitting by allowing some objects to be misclassified. Some illustrative examples, as well as a visual procedure for choosing the threshold $\beta$, based on the Receiver Operating Characteristic (ROC) curves are given in Section 4.4, ending with some concluding remarks in Section 4.5.

## *4.2 The hard-margin approach*

In this section we address the hard-margin approach, for the case in which $I$ is separable, as introduced in Section 1.2.1. All the results in this section, holds also for the case in which an embedding $\Phi$ is applied, see Section 1.2.2 for further details, by just replacing the vector $x^u$ by its image in the feature space $E$, $\Phi(x^u)$. The soft-margin approach for dealing with the nonseparable case will be addressed in Section 4.3.

In the concept of margin on a training sample $I$, as defined in Definition 1.2, both classes are equally important and are dealt with in the same way.

Hence, different misclassification costs or known a priori probabilities cannot directly be taken into account. We now extend such definition in order to get a notion of margin which deals with both classes separately.

**Definition 4.1:** Given a training sample $I$, the *margin of* $(\omega, \beta)$ *on class* $c \in \mathcal{C}$ is the minimum margin over the margin on all objects $u \in I_c$,

$$\rho^c(\omega,\beta) = \min_{u \in I_c} \rho^u(\omega,\beta) = \min_{u \in I_c} \max \left\{ \frac{c^u(\omega^\top x^u + \beta)}{\|\omega\|^\circ}, 0 \right\}, \quad c \in \{-1,1\}.$$

In the classical SVM approach, Problem (1.3) is solved to get the classifier for which the margin on $I$ is maximal. We propose a novel approach, in which instead of maximizing the margin on $I$, we simultaneously maximize the margin on both classes as defined in Definition 4.1. In addition, we impose that the hyperplane $(\omega, \beta)$ separates $I$. This yields the following biobjective optimization problem with open feasible region:

$$\begin{array}{lll} \max & \{\rho^1(\omega,\beta), \rho^{-1}(\omega,\beta)\} & \\ \text{s.t.:} & c^u\left(\omega^\top x^u + \beta\right) > 0 & \forall u \in I, \\ & \omega \in I\!R^p,\ \beta \in I\!R. & \end{array} \tag{4.1}$$

We seek the set of Pareto-optimal solutions to Problem (4.1), i.e., the set of feasible solutions $(\bar{\omega}, \bar{\beta})$ such that no $(\omega, \beta)$ exists such that

$$\begin{array}{rcl} \rho^1(\omega,\beta) & \geq & \rho^1(\bar{\omega},\bar{\beta}) \\ \rho^{-1}(\omega,\beta) & \geq & \rho^{-1}(\bar{\omega},\bar{\beta}) \end{array}$$

with at least one inequality strict. Since for $u \in I$,

$$\rho^u(\mu\omega, \mu\beta) = \rho^u(\omega,\beta) \quad \forall \mu > 0,\ \forall \omega \in I\!R^p,\ \forall \beta \in I\!R,$$

it follows for all $\mu > 0$, $\omega \in I\!R^p$, $\beta \in I\!R$, that

$$\begin{array}{rcl} \rho^1(\mu\omega,\mu\beta) & = & \rho^1(\omega,\beta) \\ \rho^{-1}(\mu\omega,\mu\beta) & = & \rho^{-1}(\omega,\beta). \end{array} \tag{4.2}$$

Hence, if $(\omega, \beta)$ is a Pareto-optimal solution of Problem (4.1), then, for any $\mu > 0$, $(\mu\omega, \mu\beta)$ is also feasible for Problem (4.1), and, by (4.2), it is also a Pareto-optimal solution of Problem (4.1).

Our final aim is to construct classifiers with adequate tradeoff of misclassification costs in the two groups *in* $\Omega$. In other words, we ideally would solve the biobjective problem

$$\begin{array}{ll} \max & \{\rho^1(\omega,\beta), \rho^{-1}(\omega,\beta)\} \\ \text{s.t.:} & c^u\left(\omega^\top x^u + \beta\right) > 0 \qquad \forall u \in \Omega, \\ & \omega \in I\!R^p,\ \beta \in I\!R. \end{array} \tag{4.3}$$

by describing the set of Pareto-optimal solutions.

Since the class $c^u$ of $u \in \Omega$ is known only for the objects $u \in I$, we consider Problem (4.1) as a surrogate of Problem (4.3), and thus the set of Pareto-optimal solutions of Problem (4.1) is seen as an approximation to the set of Pareto-optimal solutions of Problem (4.3).

First let us recall that $(\bar{\omega}, \bar{\beta})$ is a weakly efficient solution of Problem (4.1) if no feasible $(\omega, \beta)$ exists that is strictly better than $(\bar{\omega}, \bar{\beta})$ for both objectives, i.e.

$$\begin{array}{rcl} \rho^1(\omega,\beta) & > & \rho^1(\bar{\omega},\bar{\beta}) \\ \rho^{-1}(\omega,\beta) & > & \rho^{-1}(\bar{\omega},\bar{\beta}). \end{array} \tag{4.4}$$

We refer the reader to e.g. [28] for further details on these concepts of vector optimization.

We first characterize, for any given norm $\|\cdot\|^\circ$, the set of weakly efficient solutions (Theorem 4.4). Then, for the particular case of the Euclidean norm, we use this result to characterize the Pareto-optimal solutions.

Since all feasible solutions $(\omega, \beta)$ satisfy that $\rho^1(\omega,\beta) > 0$ and $\rho^{-1}(\omega,\beta) > 0$ one can generate all weakly efficient solutions by solving max-min type scalarizations, [28].

For the sake of completeness we state the following technical result

**Lemma 4.2:** The set of weakly efficient solutions of Problem (4.1) is obtained as the set of optimal solutions of

$$\begin{array}{ll} \max & \min\{\rho^1(\omega,\beta), \theta\rho^{-1}(\omega,\beta)\} \\ \text{s.t.:} & c^u(\omega^\top x^u + \beta) > 0 \qquad \forall u \in I, \\ & \omega \in I\!R^p,\ \beta \in I\!R. \end{array} \tag{4.5}$$

when $\theta \in (0, +\infty)$, in the sense that

1. any optimal solution of Problem (4.5) is weakly efficient for Problem (4.1),

2. for every weakly efficient solution $(\omega_\theta, \beta_\theta)$ of Problem (4.1) there exists $\theta \in (0, +\infty)$, such that $(\omega_\theta, \beta_\theta)$ is optimal for Problem (4.5).

**Proof.** First, let $\theta \in (0, +\infty)$ and let $(\bar{\omega}, \bar{\beta})$ be an optimal solution of Problem (4.5). By contradiction, suppose it is not weakly efficient for Problem (4.1), thus there is another $(\omega, \beta)$ strictly better than $(\bar{\omega}, \bar{\beta})$ in both objectives, i.e. they satisfy (4.4).

Therefore, the following inequality holds

$$\min\{\rho^1(\omega, \beta), \theta\rho^{-1}(\omega, \beta)\} > \min\{\rho^1(\bar{\omega}, \bar{\beta}), \theta\rho^{-1}(\bar{\omega}, \bar{\beta})\}$$

which contradicts the fact that $(\bar{\omega}, \bar{\beta})$ is optimal for Problem (4.5).

On the other hand, given a weakly efficient solution $(\bar{\omega}, \bar{\beta})$ of Problem (4.1), note that, since it is feasible, by the constraints of Problem (4.1), then $\rho^1(\bar{\omega}, \bar{\beta}) > 0$ and $\rho^{-1}(\bar{\omega}, \bar{\beta}) > 0$. Let $\theta$ be given by

$$\theta = \frac{\rho^1(\bar{\omega}, \bar{\beta})}{\rho^{-1}(\bar{\omega}, \bar{\beta})}.$$

The objective function of Problem (4.5) for such a $\theta$ yields

$$\min\{\rho^1(\omega, \beta), \frac{\rho^1(\bar{\omega}, \bar{\beta})}{\rho^{-1}(\bar{\omega}, \bar{\beta})}\rho^{-1}(\omega, \beta)\}$$

Now suppose $(\bar{\omega}, \bar{\beta})$ is not an optimum of Problem (4.5), thus there exists $(\hat{\omega}, \hat{\beta})$ feasible for Problem (4.5) and hence also for Problem (4.1) with better objective value than $(\bar{\omega}, \bar{\beta})$ ,

$$\min\{\rho^1(\hat{\omega}, \hat{\beta}), \frac{\rho^1(\bar{\omega}, \bar{\beta})}{\rho^{-1}(\bar{\omega}, \bar{\beta})}\rho^{-1}(\hat{\omega}, \hat{\beta})\} >$$

$$\min\{\rho^1(\bar{\omega}, \bar{\beta}), \frac{\rho^1(\bar{\omega}, \bar{\beta})}{\rho^{-1}(\bar{\omega}, \bar{\beta})}\rho^{-1}(\bar{\omega}, \bar{\beta})\} = \rho^1(\bar{\omega}, \bar{\beta}).$$

Therefore,

$$\rho^1(\hat{\omega}, \hat{\beta}) > \rho^1(\bar{\omega}, \bar{\beta}) \tag{4.6}$$

$$\frac{\rho^1(\bar{\omega}, \bar{\beta})}{\rho^{-1}(\bar{\omega}, \bar{\beta})}\rho^{-1}(\hat{\omega}, \hat{\beta}) > \rho^1(\bar{\omega}, \bar{\beta}). \tag{4.7}$$

The inequality (4.7) implies $\rho^{-1}(\hat{\omega}, \hat{\beta}) > \rho^{-1}(\bar{\omega}, \bar{\beta})$, which together with (4.6) contradicts the assumption that $(\bar{\omega}, \bar{\beta})$ is a weakly efficient solution.

□

For $\theta \in (0, +\infty)$, define

$$\begin{aligned} A_\theta &= \frac{2\theta}{\theta+1} \\ y_\theta^u &= \begin{cases} 1, & \text{if } u \in I_1 \\ -\theta, & \text{if } u \in I_{-1} \end{cases} \end{aligned}$$

and consider the following problem

$$\begin{array}{ll} \min & \|\omega\|^\circ \\ \text{s.t.:} & y_\theta^u(\omega^\top x^u + \beta) \geq A_\theta \qquad \forall u \in I, \\ & \omega \in I\!R^p,\ \beta \in I\!R. \end{array} \tag{$P_\theta$}$$

Observe that for $\theta = 1$, Problem $(P_1)$ coincides with Problem (1.4), which is the most used formulation for finding the optimal hard-margin hyperplane in the classical SVM. To emphasize this fact, throughout this chapter we call it Problem $(P_1)$.

**Lemma 4.3:** Given $(\bar{\omega}, \bar{\beta}) \in I\!R^p \times I\!R$, and $\theta \in (0, +\infty)$, the following statements are equivalent for $\theta \neq 1$ :

1. $(\bar{\omega}, \bar{\beta})$ is an optimal solution of Problem $(P_1)$,
2. $(\bar{\omega}, \bar{\beta} + \frac{\theta-1}{\theta+1})$ is an optimal solution of Problem $(P_\theta)$,

**Proof.** First of all, we show that some $(\bar{\omega}, \bar{\beta})$ is a feasible solution of Problem $(P_1)$, iff $(\bar{\omega}, \bar{\beta} + \frac{\theta-1}{\theta+1})$ is a feasible solution of Problem $(P_\theta)$.

Suppose $(\bar{\omega}, \bar{\beta})$ is a feasible solution of Problem $(P_1)$. For any $u \in I_{-1}$, since $\bar{\omega}^\top x^u + \bar{\beta} \geq 1$, we get

$$\bar{\omega}^\top x^u + \bar{\beta} + \frac{\theta-1}{\theta+1} \geq 1 + \frac{\theta-1}{\theta+1} = \frac{2\theta}{\theta+1} = A_\theta.$$

For $u \in I_{-1}$, feasibility of $(\bar{\omega}, \bar{\beta})$ implies $\bar{\omega}^\top x^u + \bar{\beta} \leq -1$, yielding

$$\theta(\bar{\omega}^\top x^u + \bar{\beta} + \frac{\theta-1}{\theta+1}) \leq \theta(-1 + \frac{\theta-1}{\theta+1}) = \frac{-2\theta}{\theta+1} = -A_\theta.$$

Hence, $(\bar{\omega}, \bar{\beta} + \frac{\theta-1}{\theta+1})$ is a feasible solution of Problem $(P_\theta)$.

Analogously, if $(\bar{\omega}, \bar{\beta} + \frac{\theta-1}{\theta+1})$ is a feasible solution of Problem $(P_\theta)$, then,

$$\begin{aligned} \bar{\omega}^\top x^u + \bar{\beta} = \bar{\omega}^\top x^u + \bar{\beta} + \tfrac{\theta-1}{\theta+1} - \tfrac{\theta-1}{\theta+1} \geq A_\theta - \tfrac{\theta-1}{\theta+1} = 1, \qquad & \forall u \in I_1, \\ \bar{\omega}^\top x^u + \bar{\beta} = \bar{\omega}^\top x^u + \bar{\beta} + \tfrac{\theta-1}{\theta+1} - \tfrac{\theta-1}{\theta+1} \leq \tfrac{-A_\theta}{\theta} - \tfrac{\theta-1}{\theta+1} = -1, \qquad & \forall u \in I_{-1}, \end{aligned}$$

so $(\bar{\omega}, \bar{\beta})$ is a feasible solution of Problem $(P_1)$.

Now we prove the optimality of such solutions. Given $(\bar{\omega}, \bar{\beta})$, optimal solution of Problem $(P_1)$, let $(\omega^*, \beta^*)$ be an optimal solution for Problem $(P_\theta)$. Using what we have proven above, $(\bar{\omega}, \bar{\beta} + \frac{\theta-1}{\theta+1})$ is feasible for Problem $(P_\theta)$, so $\|\omega^*\|^\circ \leq \|\bar{\omega}\|^\circ$. On the other hand, $(\omega^*, \beta^* + \frac{\theta-1}{\theta+1})$ is feasible for Problem $(P_1)$, which yields $\|\omega^*\|^\circ \geq \|\bar{\omega}\|^\circ$. Hence $(\bar{\omega}, \bar{\beta} + \frac{\theta-1}{\theta+1})$ is an optimal solution of Problem $(P_\theta)$. The other implication is analogous.

□

**Theorem 4.4:** Let $\mathcal{O}$ be the set of optimal solutions of Problem $(P_1)$. The set $\mathcal{E}$ of weakly efficient solutions of the biobjective Problem (4.1) is given by

$$\mathcal{E} = \{(\mu\omega_1, \mu\beta) : |\beta - \beta_1| < 1,\ \mu > 0,\ (\omega_1, \beta_1) \in \mathcal{O}\}.$$

**Proof.** Let $(\bar{\omega}, \bar{\beta}) \in \mathbb{R}^p \times \mathbb{R}$. By Lemma 4.2, $(\bar{\omega}, \bar{\beta})$ is weakly efficient for Problem (4.1) if and only if there exists $\theta \in (0, +\infty)$ such that $(\bar{\omega}, \bar{\beta})$ is an optimal solution of Problem (4.5). This is equivalent to $(\bar{\omega}, \bar{\beta})$ being optimal for

$$\begin{array}{lll} \min & \dfrac{\|\omega\|^\circ}{\min\left\{\min_{u\in I_1} c^u(\omega^\top x^u + \beta), \theta \min_{u\in I_{-1}} c^u(\omega^\top x^u + \beta)\right\}} & \\ \text{s.t.:} & c^u(\omega^\top x^u + \beta) > 0 & \forall u \in I, \\ & \omega \in \mathbb{R}^p,\ \beta \in \mathbb{R}. & \end{array} \tag{4.8}$$

Observe that $(\omega, \beta)$ is optimal for Problem (4.8) if and only if $(\mu\omega, \mu\beta)$ is optimal for Problem (4.8) for any $\mu > 0$. Hence, by normalizing the denominator in the objective of Problem (4.8) we have that $(\bar{\omega}, \bar{\beta})$ is optimal for Problem (4.8) if and only if there exist $\theta \in (0, +\infty)$ and $\mu > 0$ such that $(\mu\bar{\omega}, \mu\bar{\beta})$ is optimal for the following problem:

$$\begin{array}{ll} \min & \|\omega\|^\circ \\ \text{s.t.:} & \min\left\{\min_{u\in I_1}(\omega^\top x^u + \beta), \theta \min_{u\in I_{-1}}(-\omega^\top x^u - \beta)\right\} = A_\theta, \\ & \omega \in \mathbb{R}^p,\ \beta \in \mathbb{R}, \end{array}$$

where $A_\theta = \frac{2\theta}{\theta+1}$. Such a problem is equivalent to the following one

$$\begin{array}{ll} \min & \|\omega\|^\circ \\ \text{s.t.:} & \min\left\{\min_{u\in I_1}(\omega^\top x^u + \beta), \min_{u\in I_{-1}} \theta(-\omega^\top x^u - \beta)\right\} \geq A_\theta, \\ & \omega \in I\!R^p,\ \beta \in I\!R, \end{array}$$

which can be rephrased as

$$\begin{array}{lll} \min & \|\omega\|^\circ & \\ \text{s.t.:} & y_1^u(\omega^\top x^u + \beta) \geq A_\theta & \forall u \in I_1, \\ & \theta y_1^u(\omega^\top x^u + \beta) \geq A_\theta & \forall u \in I_{-1}, \\ & \omega \in I\!R^p,\ \beta \in I\!R. & \end{array} \tag{4.9}$$

Since

$$\begin{aligned} y_1^u &= y_\theta^u, \quad \forall u \in I_1 \\ \theta y_1^u &= y_\theta^u, \quad \forall u \in I_{-1}, \end{aligned}$$

Problem (4.9) is actually Problem $(P_\theta)$. Hence, $(\bar{\omega}, \bar{\beta})$ is weakly efficient iff there exists $\mu > 0$ such that $(\mu\bar{\omega}, \mu\bar{\beta})$ solves $(P_\theta)$ for some $\theta \in (0, +\infty)$. By Lemma 4.3, this is equivalent to $(\mu\bar{\omega}, \mu\bar{\beta} - \frac{\theta-1}{\theta+1})$ being optimal for $(P_1)$. Hence, $(\bar{\omega}, \bar{\beta})$ has the form $(\mu\omega_1, \mu\beta)$ with $|\beta - \beta_1| < 1$, and every $(\bar{\omega}, \bar{\beta})$ with such a form is weakly efficient for Problem (4.1). □

This result sets that, for an arbitrary norm, the classification rules given by the weakly efficient solutions of Problem (4.1) correspond to hyperplanes that are parallel to the hyperplane described by one of the classical SVM classification rules, obtained by solving Problem (1.4), as stated in Chapter 1. For the particular case of the Euclidean norm we have that all these weakly efficient solutions are also Pareto-optimal.

**Lemma 4.5:** For the particular case in which $\|\cdot\|^\circ$ is the Euclidean norm, then Problem $(P_\theta)$ has a unique solution.

**Proof.** Note that, for the Euclidean norm, $(P_\theta)$ can be rephrased as

$$\begin{array}{lll} \min & \|\omega\|_2^2 & \\ \text{s.t.:} & y_\theta^u(\omega^\top x^u + \beta) \geq A_\theta & \forall u \in I, \\ & \omega \in I\!R^p,\ \beta \in I\!R. & \end{array} \tag{4.10}$$

Given $\theta$, the function $\omega \mapsto \|\omega\|_2^2$ is strictly convex, so there exists a unique $\omega_\theta$ such that any optimal solution $(\omega, \beta)$ for Problem (4.10) has $\omega = \omega_\theta$. We show now that the set of optimal solutions of Problem (4.10) is a singleton.

For Problem (4.10), KarushKuhnTucker (KKT) conditions at $(\omega_\theta, \beta)$, which are necessary and sufficient for optimality, are given by

$$\begin{array}{ll} \lambda_\theta^u \geq 0 & \forall u \in I \\ 2\omega_\theta - \sum_{u \in I} \lambda_\theta^u y_\theta^u x^u = 0 & \\ \sum_{u \in I} \lambda_\theta^u y_\theta^u = 0 & \\ \lambda_\theta^u [y_\theta^u (\omega_\theta^\top x^u + \beta) - A_\theta] = 0 & \forall u \in I. \end{array}$$

First of all, note that $\lambda_\theta \neq 0$. Indeed, if $\lambda_\theta^u = 0$ for all $u \in I$, one would have $\omega_\theta = 0$, which simultaneously implies, since $(\omega_\theta, \beta)$ is feasible, that $\beta \geq A_\theta > 0$ and $\beta \leq \frac{A_\theta}{-\theta} < 0$. This is a contradiction, and hence $\lambda_\theta \neq 0$.

Hence, for any $(\omega_\theta, \beta)$, optimal for Problem (4.10) there exists $u \in I$ such that

$$y_\theta^u (\omega_\theta^\top x^u + \beta) - A_\theta = 0$$

i.e.:

$$\beta = \frac{A_\theta}{y_\theta^u} - \omega_\theta^\top x^u. \tag{4.11}$$

This means that the set of optimal solutions of Problem (4.10) is finite. On the other hand, by convexity, for any two different optimal solutions of Problem (4.10), all the solutions in the segment between them are optimal. This contradicts the finiteness of the set of optimal solutions of Problem (4.10), yielding the conclusion that such a set has an unique solution, $(\omega_\theta, \beta_\theta)$, with $\beta_\theta$ of the form (4.11) for some $u \in I$.

□

**Corollary 4.6:** For the particular case in which $\|\cdot\|^\circ$ is the Euclidean norm, then the set of Pareto-optimal solutions of the biobjective Problem (4.1) is given by $\mathcal{E}$,

$$\mathcal{E} = \{(\mu\omega_1, \mu\beta) : |\beta - \beta_1| < 1, \mu > 0, (\omega_1, \beta_1)\},$$

where $(\omega_1, \beta_1)$ is the unique optimal solution of Problem $(P_1)$.

**Proof.** Any Pareto-optimal solution is, by definition, weakly efficient. Let us show the converse. Let $(\bar{\omega}, \bar{\beta})$ be weakly efficient. By Lemma 4.3 and

Theorem 4.4, there exist $\theta \in (0, +\infty)$ and $\mu > 0$ such that $(\mu\bar{\omega}, \mu\bar{\beta})$ solves $(P_\theta)$.

Suppose $(\bar{\omega}, \bar{\beta})$ is not Pareto-optimal. Then $(\mu\bar{\omega}, \mu\bar{\beta})$ would not be Pareto-optimal either. Hence there would exist $(\omega', \beta')$ such that

$$\begin{array}{rcl} \rho^1(\omega', \beta') & \geq & \rho^1(\mu\bar{\omega}, \mu\bar{\beta}) = \rho^1(\bar{\omega}, \bar{\beta}), \\ \rho^{-1}(\omega', \beta') & \geq & \rho^{-1}(\mu\bar{\omega}, \mu\bar{\beta}) = \rho^{-1}(\bar{\omega}, \bar{\beta}), \end{array} \tag{4.12}$$

with at least one of those inequality strict.

Without loss of generality we can assume that $\|\omega'\|_2 = \|\mu\bar{\omega}\|_2$. Then (4.12) is equivalent to

$$\begin{array}{rcl} \min_{u \in I_1} y_\theta^u(\omega'^\top x^u + \beta') & \geq & \min_{u \in I_1} y_\theta^u(\mu\bar{\omega}^\top x^u + \mu\bar{\beta}) \geq A_\theta, \\ \min_{u \in I_{-1}} y_\theta^u(\omega'^\top x^u + \beta') & \geq & \min_{u \in I_{-1}} y_\theta^u(\mu\bar{\omega}^\top x^u + \mu\bar{\beta}) \geq A_\theta. \end{array}$$

Hence, $(\omega', \beta')$ would be feasible for Problem $(P_\theta)$. Since its objective value at $(\omega', \beta')$ is $\|\omega'\|_2^2 = \|\mu\bar{\omega}\|_2^2$, we would have that $(\omega', \beta')$ is also optimal for Problem $(P_\theta)$. By Lemma 4.5, $(P_\theta)$ has a unique optimal solution. Thus $(\omega', \beta') = (\mu\bar{\omega}, \mu\bar{\beta})$, contradicting that at least one of the inequalities in (4.12) is strict.

□

## 4.3 The soft-margin approach

When the set $I$ is not linearly separable, no hyperplane exists classifying correctly all data points, and thus Problem (1.4) is infeasible. For these cases, the hard-margin approach can be extended to the so-called soft-margin approach, see Section 1.2.3 for a description, which consists of allowing some objects in $I$ to be misclassified, by perturbing Problem (1.4) in order to make it feasible, as described in Section 1.2.3. In this section, we see that, for the case in which the amount $\|\omega\|^\circ + C\|\xi\|^*$ defines a norm $\|\cdot\|'$ in $\mathbb{R}^p \times \mathbb{R}^{\sharp(I)}$, the results in Section 4.2 can be extended for the soft-margin approach.

Let $\|\cdot\|'$ be a norm in $\mathbb{R}^p \times \mathbb{R}^{\sharp(I)}$. Then, a general soft-margin maximization problem can be formulated as:

$$\begin{array}{lll} \min & \|(\omega, \xi)\|' & \\ \text{s.t.:} & c^u\left(\omega^\top x^u + \beta\right) + \xi^u \geq 1 & \forall u \in I, \\ & \omega \in \mathbb{R}^p,\ \beta \in \mathbb{R},\ \xi \in \mathbb{R}^{\sharp(I)}. & \end{array} \tag{4.13}$$

Common choices for $\|\cdot\|'$ are, for instance:

- $\|(\omega,\xi)\|' = \|\omega\|_1 + C\|\xi\|_1$, with $\|\cdot\|_1$ being the $L_1$ norm, which yields Problem (1.10).
- $\|(\omega,\xi)\|' = \sqrt{(\|\omega\|_2)^2 + C(\|\xi\|_2)^2}$.

$C$ is a constant which is usually chosen by crossvalidation techniques, see e.g. [26, 71, 72] and is used in order to tradeoff the perturbations $\xi^u$ and the classification scores $\omega^\top x^u + \beta$.

Moreover, we can follow [69], where a more general approach is proposed, in which the perturbations are weighed by different parameters $C_1$ and $C_{-1}$, yielding the problem

$$\begin{array}{lll} \min & \|\omega\|_2^2 + C_1 \sum_{u\in I_1}(\xi^u)^2 + C_{-1}\sum_{u\in I_{-1}}(\xi^u)^2 & \\ \text{s.t.:} & c^u\left(\omega^\top x^u + \beta\right) + \xi^u \geq 1 & \forall u \in I, \\ & \omega \in I\!R^p,\ \beta \in I\!R,\ \xi \in I\!R^{\sharp(I)}, & \end{array} \tag{4.14}$$

which corresponds to $\|\cdot\|'$ equal to

$$\|(\omega,\xi)\|' = \sqrt{\|\omega\|_2^2 + C_1 \sum_{u\in I_1}(\xi^u)^2 + C_{-1}\sum_{u\in I_{-1}}(\xi^u)^2}. \tag{4.15}$$

The parameters $C_1$ and $C_{-1}$ allow the incorporation of different a priori probabilities or misclassification costs in an approximate way, [41]. The class $c$ having smaller a priori probability (or classification cost) should have the large $C_c$ value. For instance in [41] $C_c = \frac{1}{n_c}$, where $n_c$ denotes the number of objects in $I_c$, for $c \in \{1,-1\}$ is suggested. With this, a priori probabilities, as well as different misclassification costs for each class, can be taken into account to weigh the perturbations, but not the margin itself, which is the main aim of this chapter.

In order to characterize the set of weakly efficient solutions of Problem (4.13), we reduce ourselves to the separable case as follows: first of all, we define an embedding $\Phi : I\!R^p \to I\!R^p \times I\!R^{\sharp(I)}$, having two components: the first component is the predictor vector $x$, and the second component is a vector $\delta$ given by

$$\delta_u(x) = \begin{cases} 1 & \text{if there exists } u \in I_1 \text{ with } x = x^u \\ -1 & \text{if there exists } u \in I_{-1} \text{ with } x = x^u \\ 0 & \text{otherwise.} \end{cases}$$

In the embedded feature space $\mathbb{R}^p \times \mathbb{R}^{\sharp(I)}$, the score function (1.1) becomes

$$f(x) = (\omega, \xi)^\top (x^u, \delta(x^u)) + \beta.$$

Note that, for $u \in I_1$, all the components of $\delta(x^u)$ are zero, but the $u-$th component which is equal to one. Analogously, for $u \in I_{-1}$, all the components of $\delta(x^u)$ are zero, but the $u-$th component which is equal to $-1$. Hence, after the embedding $\Phi(x) = (x, \delta(x))$, the margin of $(\omega, \xi, \beta)$ on class $c$, defined in Definition 4.1, is given by

$$\rho^c(\omega, \xi, \beta) = \min_{u \in I_c} \rho^u(\omega, \xi, \beta) = \min_{u \in I_c} \max \left\{ \frac{c^u((\omega, \xi)^\top (x^u, \delta(x^u)) + \beta)}{\|(\omega, \xi)\|'}, 0 \right\}.$$

As in Section 4.2, we now consider the problem of simultaneously maximizing the margin on both classes, imposing that $(\omega, \xi, \beta)$ separates $I$ :

$$\begin{array}{lll} \max & \{\rho^1(\omega, \xi, \beta), \rho^{-1}(\omega, \xi, \beta)\} & \\ \text{s.t.:} & c^u \left(\omega^\top x^u + \beta\right) + \xi^u > 0 & \forall u \in I, \\ & \omega \in \mathbb{R}^p, \beta \in \mathbb{R}, \xi \in \mathbb{R}, & \end{array} \tag{4.16}$$

Applying Theorem 4.4 we get a characterization of the set $\hat{\mathcal{E}}$ of the weakly efficient solutions of Problem (4.16):

$$\hat{\mathcal{E}} = \{(\mu\omega_1, \mu\xi_1, \mu\beta) : |\beta - \beta_1| < 1, \mu > 0, (\omega_1, \xi_1, \beta_1) \in \hat{O}\}.$$

where $\hat{O}$ denotes the set of optimal solutions of Problem (4.13). Moreover, for the case in which the norm $\|\cdot\|'$ is given by (4.15), we have a Corollary analogous to Corollary 4.6.

**Corollary 4.7:** For the particular case in which $\|\cdot\|'$ is the norm given by (4.15), then the set of Pareto-optimal solutions of the biobjective Problem (4.16) is given by $\hat{\mathcal{E}}$,

$$\hat{\mathcal{E}} = \{(\mu\omega_1, \mu\xi_1, \mu\beta) : |\beta - \beta_1| < 1, \mu > 0\},$$

where $(\omega_1, \xi_1, \beta_1)$ is the unique optimal solution of Problem (4.14).

Recall that, for $\mu > 0$, the classification rules given by $(\mu\omega, \mu\xi, \mu\beta)$ are equivalent in the sense that they allocate objects in exactly the same way. As a consequence, all the classification rules corresponding to Pareto-optimal solutions of Problem (4.16) are given by moving the parameter $\beta$ in the classification rule obtained by solving Problem (4.14) and all such solutions corresponds to parallel hyperplanes.

## 4.4 Illustrative examples

As has been proven in the preceding sections, the best classification rules, in the Pareto sense, are obtained by varying the value $\beta$ in the optimal solutions of the classical SVM. The choice of such a value $\beta$ is up to the decision maker, who should take into account the tradeoff between the misclassification costs and a priori probabilities in both classes.

In order to choose a value for the parameter $\beta$, some authors (see e.g. [48]) have suggested the use of the ROC curve. The ROC curve shows the *sensitivity*, i.e. the proportion of correctly classified objects of the positive class, against the *specificity*, proportion of correctly classified objects of the negative class, for different values of the parameter $\beta$. The ROC curves can help the decision maker in the choice of $\beta$, since that is the only free parameter, as shown by the characterization given in Corollary 4.7.

In order to show how to guide the choice of $\beta$ in real-life settings, we have performed various experiments using those databases in the UCI Machine Learning Repository having two classes and no missing data whose predictor variables are all continuous, as detailed in the summary table available in [6]: `bupa`, `ionosphere`, `pima`, `sonar` and `wdbc`. A brief description of them is presented in Appendix A. From each database, a random sample of 100 objects is drawn and used as training sample $I$ and the remaining is used as testing sample in order to validate the model.

All the numerical results have been performed by using the SVM toolbox for Matlab [62]. Data were not preprocessed and a linear kernel was used in all the experiments. The norm $\|\cdot\|'$ used in the experiments is given by (4.15). The parameters $C_1$ and $C_{-1}$ where set to be equal, and their value chosen by crossvalidation, as implemented in the popular SVM library LIBSVM [20].

With this information at hand, we can draw the ROC curve *in the training sample*, i.e. the plot, when $\beta$ varies, of the proportions of misclassified objects in both classes in the available set of data. This is not the ROC curve for the whole population, which is unknown in real applications. We then use the former as a surrogate of the latter. In Figures 4.1-4.5 we give the ROC curves for the training sample (thick lines) and testing sample (thin lines). The SVM solution is marked with a star. However, it is not evident to see from ROC curves the effect of the $\beta$ in the tradeoff between sensitivity and specificity, since the value $\beta$ yielding each pair is not plotted.

In Figures 4.6-4.10, specificity and sensitivity are shown for both training and testing sample (training in thick) in such a way that the decision maker

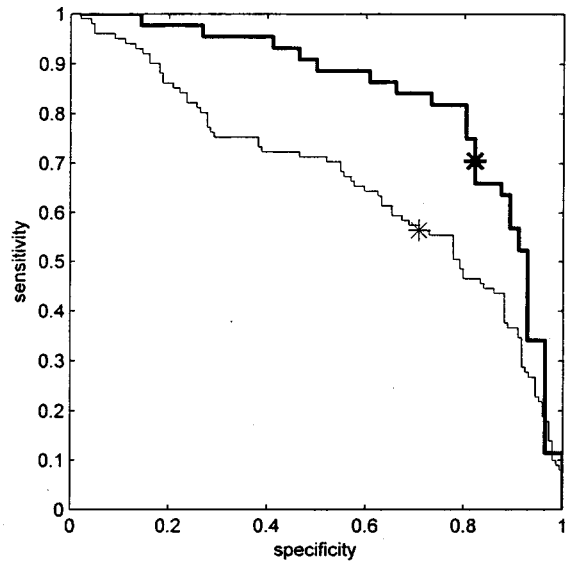


*Fig. 4.1:* ROC curve. Database: `bupa`. C=0.03125

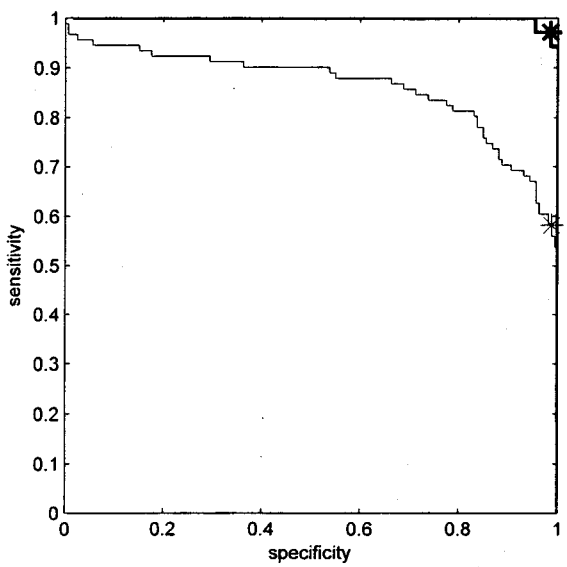


*Fig. 4.2:* ROC curve. Database: `ionosphere`. C=2.0

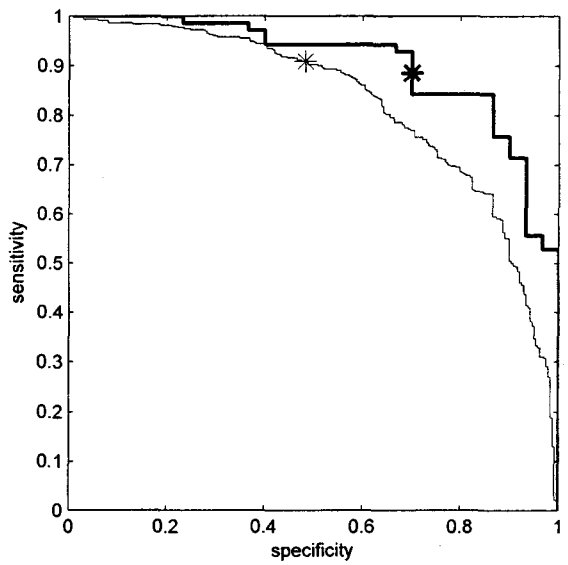


*Fig. 4.3:* ROC curve. Database: `pima`. C=0.03125

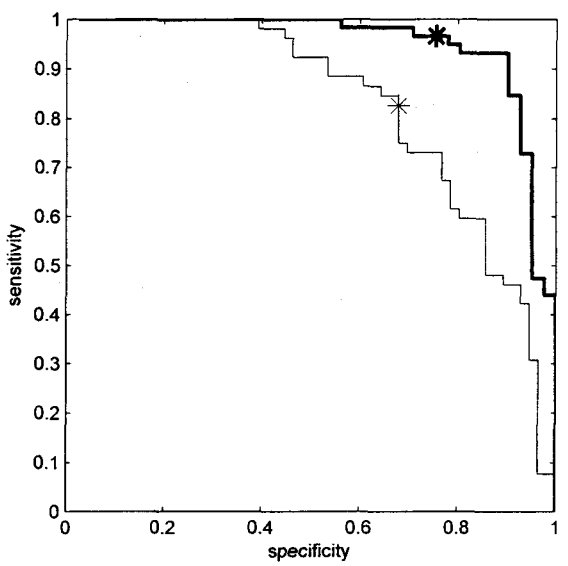


*Fig. 4.4:* ROC curve. Database: `sonar`. C=0.5

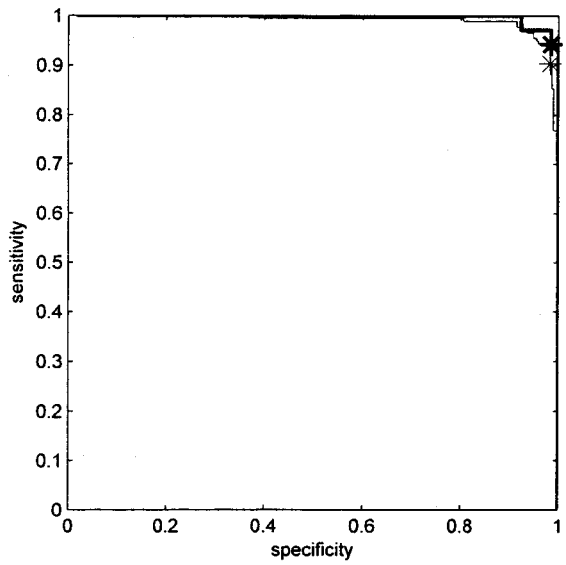


*Fig. 4.5:* ROC curve. Database: `wdbc`. C=2.0

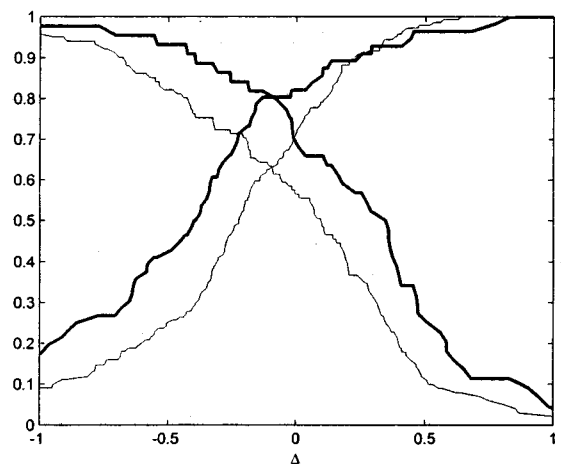


*Fig. 4.6:* Specificity and sensitivity of $(\omega_1, \beta_1 - \Delta)$, for a threshold $\Delta$ and $(\omega_1, \beta_1)$ optimal solution of (4.14). Database: `bupa`. C=0.03125

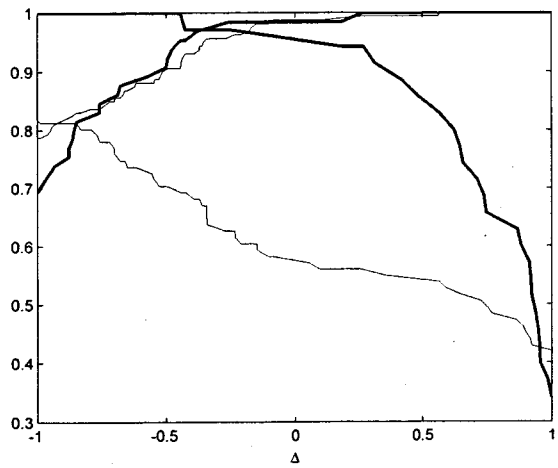


*Fig. 4.7:* Specificity and sensitivity of $(\omega_1, \beta_1 - \Delta)$, for a threshold $\Delta$ and $(\omega_1, \beta_1)$ optimal solution of (4.14). Database: `ionosphere`. C=2.0

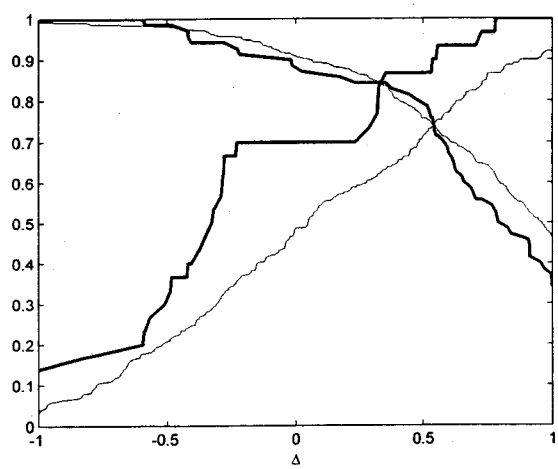


*Fig. 4.8:* Specificity and sensitivity of $(\omega_1, \beta_1 - \Delta)$, for a threshold $\Delta$ and $(\omega_1, \beta_1)$ optimal solution of (4.14). Database: `pima`. C=0.03125

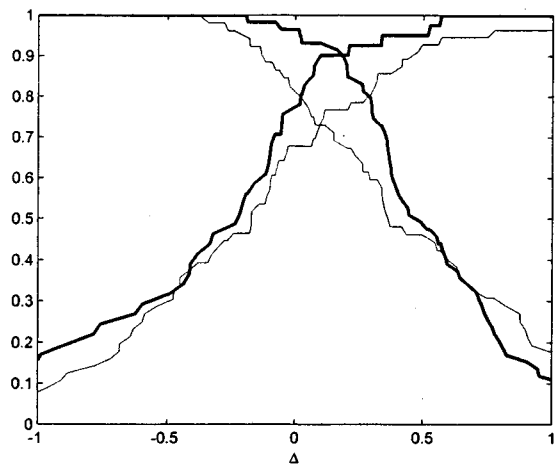


*Fig. 4.9:* Specificity and sensitivity of $(\omega_1, \beta_1 - \Delta)$, for a threshold $\Delta$ and $(\omega_1, \beta_1)$ optimal solution of (4.14). Database: `sonar`. C=0.5

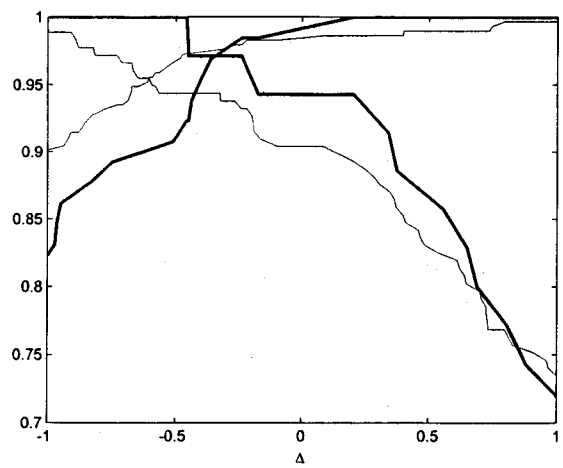


*Fig. 4.10:* Specificity and sensitivity of $(\omega_1, \beta_1 - \Delta)$, for a threshold $\Delta$ and $(\omega_1, \beta_1)$ optimal solution of (4.14). Database: `wdbc`. C=2.0

can choose a value for the parameter $\beta$. Sensitivity and specificity values for the classical SVM correspond to the case $\Delta = 0$. The higher $\Delta$, the higher $\rho^{-1}(\omega, \beta, \xi)$, at the expense of decreasing $\rho^{1}(\omega, \beta, \xi)$. This, as empirically illustrated in the graphics, translates into saying that the higher the value chosen for $\Delta$, the higher the specificity and the lower the sensitivity.

## 4.5 Conclusions

In this chapter, the concept of margin in a training sample $I$ has been generalized to the margin in a class, in order to deal separately with them via a Biobjective Program. Then, for any norm $\|\cdot\|^{\circ}$, the set of hyperplanes which are weakly efficient for the problem of simultaneously maximazing the margin on both classes, has been characterized. Moreover, it has been proven that, for the particular case of the Euclidean norm, the set of hyperplanes which are Pareto-optimal in the simultaneous optimization of the margin in both classes, is given by a set of parallel hyperplanes, one of which is just the optimal margin hyperplane as defined by the correspondent unidimensional SVM.

This chapter proposes a simple way for taking into account different misclassification costs, or known a priori probabilities of the classes. Our main result gives a theoretical foundation for the commonly used ROC approach for tuning the parameter $\beta$.

The equivalence between the set of weakly efficient solutions and the set of Pareto-optimal solutions has been proven for the Euclidean case. We are exploring now how to characterize the set of Pareto-optimal solution when a polyhedral norm, like the $L_1$ norm, is used. This case differs from the Euclidean case mainly because the solution of the classical one-objective SVM is not unique.

We have dealt in this work with the two-group case. Extending it to the case in which more than two classes exists is not trivial and it is a promising issue for further research.

# 5. MULTIGROUP SVM WITH MEASUREMENT COSTS

## *5.1 Introduction*

Correctly predicting the class membership of an object has been our main goal throughout this thesis. However, in real-word classification problems it is very convenient to obtain classification rules that, not only achieve good classification behavior, but are also cheap or quick. A typical example is medical diagnosis, where some tests are much more expensive or take much longer than others. For instance, in a poll by KDnuggets [45], 'dealing with unbalanced and cost-sensitive data' was selected, by both practitioners and academics, between the most important Data Mining topics, where cost-sensitive data means data 'which has different cost to get it,' [46].

If the classification rule does not use variables based on the most expensive tests, classifying new patients will be much cheaper or quicker, and hopefully without deteriorating significantly the quality of classification.

In Chapter 4, we have analyzed the case in which there exists a cost associated to misclassification of objects depending on their class. Together with this type of cost, which is related with the classification ability of the rule, other costs, linked to the variables or attributes, can be defined. In the simplest model we associate equal costs to each predictor variable. In that case, keeping the total cost below a given level amounts to stating an upper bound on the number of variables to be used. This, called feature selection, is a well-known problem in the Data Mining literature, see e.g. [35] for an introduction. Turney [66] proposed other types of nontrivial costs, for instance the test cost, also called measurement cost, where each test (attribute, measurement, feature) has an associated cost, such as economical payment, risk incurred by the act of measuring it, computational effort or some kind of complexity. The aim of minimizing such costs has been mentioned before in the literature as a desirable consequence of feature selection, see e.g. [35], but hardly directly addressed.

In this chapter, we address classification problems in which both misclas-

sification rate and measurement costs are relevant. To do this, we formulate the Biobjective Program, see e.g. [28], of the simultaneous minimization of the misclassification rate, via the maximization of the margin, and the measurement costs. Pareto-optimal solutions, i.e. classifiers that cannot be improved at the same time in both objectives, are sought. The set of Pareto-optimal solutions of the Biobjective Program gives us a finite set of classification rules, in such a way that any rule which is not Pareto-optimal should be discarded, since it is beaten in terms of margin and cost by another rule. Choosing one out of the set of Pareto-optimal rules is done by an appropriate compromise between the two criteria involved.

We have structured the chapter as follows. The classification model followed in this chapter is presented in Section 5.2. The measurement costs we want to incorporate are modeled in Section 5.3. Formulations for the hard-margin maximization problem are derived in Section 5.4. Costs and margin are incorporated in a Biobjective problem described in Section 5.5 and a soft-margin version of it is proposed in Section 5.6. Section 5.7 is devoted to some numerical results and, some conclusions are presented in Section 5.8.

## 5.2 The classification model

We are considering in this chapter the general case $\mathcal{C} = \{1, 2, \ldots, Q\}$ Hence, we consider the multigroup SVM approach described in Section 1.3. We work with a score function $f$ given by (1.17) where an embedding $\Phi = (\phi_1, \phi_2, \ldots, \phi_N) : I\!R \to E$ is given. Denote by $\mathcal{F}$ the family of all features $\phi_k$ for $k = 1, 2, \ldots, N$. For instance, family $\mathcal{F}$ given by

$$\mathcal{F} = \{x_1, x_2, \ldots, x_p\}, \tag{5.1}$$

yields linear classifiers, as in (1.17), whereas quadratic classifiers, [25, 29], are obtained by setting

$$\mathcal{F} = \{x_1, x_2, \ldots, x_p\} \bigcup \{x_i x_j : 1 \leq i \leq j \leq p\} \tag{5.2}$$

i.e., the set of monomials of degree up to 2.

This framework also includes voting classifiers, such as boosting, e.g. [24, 31], in which $\mathcal{C} = \{-1, 1\}$ and a set of primitive classifiers $\phi_k : X \to \{0, 1\}$

$$\phi_k(x) = 1 \text{ iff } x \text{ is allocated to class 1 via the } k\text{-th classifier},$$

are combined linearly into a single score function of the form (1.5). For a very promising strategy for generating such primitive classifiers see e.g. [7].

## 5.3 Measurement costs

As stated in Chapter 1, finding a score function $f$ as in (1.17) which separates the groups and has maximal margin (either in its hard or soft versions) is a plausible criterion when obtaining the predictor vector $x^u$ is costless. When this is not the case, we should also take into account the cost associated with the evaluation of the classification rule.

In many practical applications, as medical diagnosis, the predictor variables of the data may be some diagnosis test (such as blood test, ...) that have associated a cost, either money, or risk/damage incurred to the patient. If the classifier built does not depend on some of these variables, we could avoid their measurement (and the corresponding cost) in the diagnosis of new patients. In this situation, we should seek a classifier that enjoys good generalization properties, and at the same time, has low cost.

Obtaining cheaper or quicker classification rules have been mentioned as one of the desirable consequences of feature selection, where the aim is to reduce the number of variables or features used by the classification rule. However costs associated with such variables or features have seldom been considered.

Several authors have addressed measurement cost issues related with classification. For instance, [53, 54, 65] consider classification trees whose branching rule takes such costs into account. See [66] for a comparison of such methods and [4, 66] and the references therein for other proposals. In most cases, the unique goal is to minimize some surrogate of the expected misclassification cost, and, since the algorithm takes somehow into account measurement costs, it is hoped that the rule obtained this way has a low associated measurement cost.

In this chapter, however, we explicitly consider the minimization of measurement costs as one criterion, whose trade-off with margin optimization is to be determined by the user.

Costs are modeled as follows: Denote by $\Pi_k$ the cost associated with evaluating the feature $\phi_k \in \mathcal{F}$ at a given $x$. For instance, if we are following a linear approach, as given by (5.1), $\Pi_l$ represents the cost of measuring the predictor variable $l$ in a new object.

Given the parameter $W = (\omega^1, \ldots, \omega^Q)$, define

$$S(W) = \{k \mid \exists c \in \mathcal{C} : \omega_k^c \neq 0,\ 1 \leq k \leq N\}.$$

In other words, $S(W)$ represents the set of features needed to classify new objects. In principle, these are the features we have to pay for, so a score function $f$ with coefficients $(W, b)$ will have associated a measurement cost equal to

$$\pi(W, b) = \sum_{k \in S(W)} \Pi_k. \tag{5.3}$$

Pure linearity, as assumed in (5.3), may be unrealistic in some practical situations. For instance, it may be the case that, once we have incurred a cost for obtaining some feature $\phi_k$, some other features may be given for free or at reduced cost. This may happen, for example, in a medical context when the measurement of a variable requires a blood extraction, and some other variables can be measured using the same blood test. Another context where one encounters this, is the case in which some features are functions of other features: In model (5.2), feature $\Phi(x) = x_i x_j$ is obtained for free if both features $\Phi(x) = x_i$ and $\Phi(x) = x_j$ have been previously inspected.

**Example 5.1:** In Table 5.1 one can see the costs of a simple example with two classes $Q = 2$, and $\mathcal{F} = \{\phi_1, \ldots, \phi_5\}$ with different costs. The score function given by $f_1 = \phi_1 + 4\phi_5$ and $f_2 = 3\phi_1 + 2$ incurs a cost of $2 + 2 = 4$.

| features | $\phi_1$ | $\phi_2$ | $\phi_3$ | $\phi_4$ | $\phi_5$ |
|---|---|---|---|---|---|
| costs | 2 | 5 | 3 | 0 | 2 |

*Tab. 5.1:* Example of feature cost

In order to consider situations like the one described in Example 5.1, suppose that precedence constraints, in the form of a partial order $\prec$ between the features, are given. This means that if $h \preceq k$, the use of the feature $\phi_k$ requires also the payment for feature $\phi_h$. Moreover, in computing the total cost, the cost for every feature has to be summed at most once. In order to formalize this, define an auxiliary variable $z_k \in \{0, 1\}$ for each $k = 1, \ldots, N$, representing

$$z_k = \begin{cases} 1 & \text{if payment of } \Pi_k \text{ is needed} \\ 0 & \text{otherwise} \end{cases}$$

in other words:

$$z_k = \begin{cases} 1 & \text{if } h \in S(W) \text{ for some } h \text{ with } k \preceq h \\ 0 & \text{otherwise.} \end{cases}$$

Thus, cost associated with a score function with coefficients $(W, b)$ will be

$$\pi(W, b) = \sum_{k=1}^{N} z_k \Pi_k. \tag{5.4}$$

Particular cases already suggested in the literature can be easily accommodated into our framework. For instance, in [55] variables are grouped in a way that, if one variable from a group is requested, then all the others in the same group are available for zero additional cost. To model this case in our setting, define the cost of one variable from each group to be equal to the cost of the group it belongs to, and set the remaining variables to have zero cost. Moreover, choose a partial order $\prec$ for which $h \prec j$ iff variables $h$ and $j$ are in the same group and $h$ has nonzero cost.

Moreover, this modelling technique allows us to use, but it is not limited to, polynomial kernels. Indeed, suppose a kernel $k(x, y) = \Phi(x)^\top \Phi(y)$ for some $\Phi : X \to E$. If $\Phi$ holds

- $E$ is a finite dimensional *feature space*, $E \subset \mathbb{R}^N$,
- for any component $\phi_k$, $k = 1, 2, \ldots, N$, of $\Phi = (\phi_1, \phi_2, \ldots, \phi_N)$, the information about what original variables are needed to calculate $\phi_k$ is available,

then, the cost associated to a score function can be modeled using the methodology explained in this section. Biobjective problems for other kernels, which do not hold the conditions above, can also be formulated [74]. However, they yield combinatorial problems which are much harder to solve in practice.

We will show in Sections 5.5 and 5.6, that this modelling technique allows formulations as Biobjective Mixed Integer Programs. For these models there exist suitable techniques for finding their Pareto-optimal solutions.

Minimizing (5.4) will be one of our goals. However, our main goal is finding classifiers with good generalization properties. This, the second objective in our model, will be discussed in detail in the following section.

## 5.4 Margin optimization

As stated in Chapter 1, SVM uses the maximization of the margin as a surrogate of the classification ability to find a good classifier. We follow here the multigroup hard-margin maximization problem (1.15) described in Section 1.3.2, which, after the embedding described in Section 5.2, leads the following problem

$$\begin{array}{ll} \max & \theta(W,b) \\ \text{s.t:} & \|W\|^\circ \leq 1, \\ & (W,b) \in I\!R^{NQ} \times I\!R^{Q}. \end{array} \tag{5.5}$$

We now derive some properties of Problem (5.5) that allows us to rephrase it as an equivalent problem with less decision variables and detect the non-separability of $I$ for a given family of features $\mathcal{F}$.

**Property 5.2:** Problem (5.5) has finite optimal value.

**Proof.** Let $(W,b) = (\omega^1, \ldots, \omega^Q; \beta^1, \ldots, \beta^Q)$ be a feasible solution of Problem (5.5).

Let $u \in I$ and $j \neq c^u$, then

$$\begin{aligned} & |(\omega^{c^u})^\top \Phi(x^u) + \beta^{c^u} - (\omega^j)^\top \Phi(x^u) - \beta^j| \\ = \; & |(\omega^{c^u} - \omega^j)^\top \Phi(x^u) + \beta^{c^u} - \beta^j| \\ \leq \; & |(\omega^{c^u} - \omega^j)^\top \Phi(x^u)| + |\beta^{c^u} - \beta^j|. \end{aligned}$$

To bound the first term, observe that, since all norms are equivalent, there exists $K$ such that $|\omega_k^c| \leq K$ for all $k = 1, 2, \ldots, N$, $c \in \mathcal{C}$.

Hence,

$$\begin{aligned} & |(\omega^{c^u} - \omega^j)^\top \Phi(x^u)| \\ \leq \; & \sum_{k=1}^{N} |\omega_k^{c^u} - \omega_k^j| |\phi_k(x^u)| \\ \leq \; & 2KN \max_{1 \leq k \leq N,\, u \in I} |\phi_k(x^u)| = K' < \infty. \end{aligned}$$

Now, we will bound the term $|\beta^{c^u} - \beta^j|$. Since each class is represented, $I_j \neq \emptyset$, let $v \in I_j$. Solution $(W,b)$ feasible for Problem (5.5) implies both $u$ and $v$ are correctly classified,

$$\begin{aligned} (\omega^{c^u})^\top \Phi(x^u) + \beta^{c^u} - ((\omega^j)^\top \Phi(x^u) + \beta^j) > 0 \\ (\omega^{c^u})^\top \Phi(x^v) + \beta^{c^u} - ((\omega^j)^\top \Phi(x^v) + \beta^j) < 0 \end{aligned}$$

yielding,

$$(\omega^{c^u} - \omega^j)^\top \Phi(x^v) < \beta^j - \beta^{c^u} < (\omega^{c^u} - \omega^j)^\top \Phi(x^u).$$

Thus

$$\begin{aligned} & |\beta^j - \beta^{c^u}| \\ \leq\ & \max\{|(\omega^{c^u} - \omega^j)^\top \Phi(x^u)|, |(\omega^{c^u} - \omega^j)^\top \Phi(x^v)|\} \\ \leq\ & \max_{v \in I}\{|(\omega^{c^u} - \omega^j)^\top \Phi(x^v)|\} \\ \leq\ & 2KN \max_{1 \leq k \leq N,\, v \in I} |\phi_k(x^v)| = K'. \end{aligned}$$

Hence the objective function is bounded by

$$\theta(W, b) = \min_{u \in I} \min_{j \neq c^u} |(\omega^{c^u})^\top \Phi(x^u) + \beta^{c^u} - (\omega^j)^\top \Phi(x^u) - \beta^j| \leq 2K'.$$

□

We have assumed that $\mathcal{F}$ is rich enough to enable separability of $I$. However, it may be useful to have a method to check such separability. In case we do not know if $I$ is separable in the feature space $E$, solving Problem (5.5) allows us to check it. Indeed we have the property:

**Property 5.3:** $I$ is separable if and only if Problem (5.5) has strictly positive optimal value.

Another reduction of Problem (5.5) is even possible. Recall that, for all $\lambda \in \mathbb{R}$, the score functions defined by $(W, b)$ and $(W, \tilde{b})$, with $\tilde{b}^c = b^c + \lambda$ for all $c \in \mathcal{C}$, are equivalent in the sense that both classify objects to the same classes, and both have the same margins. Then, we can restrict the coefficients $\beta^c$ to be nonnegative, yielding the problem:

$$\begin{array}{ll} \max & \theta(W, b) \\ \text{s.t.:} & \|W\|^\circ \leq 1 \\ & (W, b) \in \mathbb{R}^{NQ} \times \mathbb{R}^Q_+. \end{array} \tag{5.6}$$

**Property 5.4:** Problems (5.5) and (5.6) are equivalent in the sense that every optimal solution of Problem (5.6) is also optimal for Problem (5.5), and, for any optimal solution of Problem (5.5), there exists a feasible solution of Problem (5.6) that is also optimal in both problems.

## 5.5 A biobjective approach

In the last sections we have described the two objectives of our problem, namely, maximizing the margin and minimizing the measurement cost. Hence we have the following biobjective problem:

$$\begin{array}{ll} \max & \theta(W,b) \\ \min & \pi(W,b) \\ \text{s.t.:} & \|W\|^\circ \leq 1 \\ & (W,b) \in I\!R^{NQ} \times I\!R^Q_+. \end{array} \tag{5.7}$$

We are seeking the set of Pareto optimal solutions of biobjective problem (5.7), i.e. those feasible solutions that cannot be improved simultaneously in both objectives.

**Property 5.5:** The set of Pareto-optimal outcomes of the biobjective problem (5.7) is finite.

**Proof.** The set of all outcomes of Problem (5.7) can be calculated by solving the problem

$$\begin{array}{ll} \max & \theta(W,b) \\ \text{s.t.:} & \|W\|^\circ \leq 1 \\ & \pi(W,b) \leq \pi \\ & (W,b) \in I\!R^{NQ} \times I\!R^Q_+ \end{array}$$

for any $\pi$ in the set of possible costs:

$$\{\pi(W,b) : (W,b) \in I\!R^{NQ} \times I\!R^Q_+\},$$

which is contained in the finite set $\left\{\sum_{k\in S} \Pi_k : S \subseteq \{1,2,\ldots,N\}\right\}$. □

Using the notation of Section 5.4, the biobjective problem (5.7) can also

be reformulated as

$$
\begin{array}{lll}
\max & y & \\
\min & \sum_{k=1}^{N} \Pi_k z_k & \\
\text{s.t.:} & \sum_{k=1}^{N} \phi_k(x^u)\left(\omega_k^i - \omega_k^j\right) + \beta^i - \beta^j - y \geq 0, & \forall i \neq j;\ i,j \in \mathcal{C},\ u \in I_i \\
& \|W\|^\circ \leq 1 & \\
& -z_k \leq \sum_{k:h \preceq k} \sum_{c=1}^{Q} \omega_k^c \leq z_h & \forall h = 1,2,\ldots,N \\
& \omega_k^c \text{ unrestricted} & \forall k = 1,2,\ldots,N;\ c \in \mathcal{C} \\
& y \text{ unrestricted} & \\
& \beta^c \geq 0 & \forall c \in \mathcal{C} \\
& z_k \in \{0,1\} & \forall k = 1,2,\ldots,N.
\end{array}
\tag{5.8}
$$

Recall that, due to the presence of a nonlinear constraint ($\|W\|^\circ \leq 1$), Problem (5.8) is a biobjective Mixed Integer Nonlinear Program. As in Chapters 2 and 3, we suggest the use of a polyhedral norm, such as, for instance, a scaled $L_1$-norm, $\|W\|_1 = \frac{1}{N}\sum_{k=1}^{N}\sum_{c=1}^{Q}|\omega_k^c|$. Then, Problem (5.7), can be rewritten as a biobjective mixed integer *linear* problem, as stated below.

**Property 5.6:** Let $\|\cdot\|^\circ$ be a scaled $L_1$-norm, $\|W\|_1 = \frac{1}{N}\sum_{k=1}^{N}\sum_{c=1}^{Q}|\omega_k^c|$. Then, Problem (5.7) can be formulated as the following Biobjective Mixed Integer Problem,

$$
\begin{array}{lll}
\max & y & \\
\min & \sum_{k=1}^{N} \Pi_k z_k & \\
\text{s.t.:} & \sum_{k=1}^{N} \phi_k(x^u)\left(\omega_{+k}^i - \omega_{-k}^i - \omega_{+k}^j + \omega_{-k}^j\right) + \beta^i - \beta^j - y \geq 0, & \\
& & \forall i \neq j;\ i,j \in \mathcal{C},\ u \in I_i \\
& \sum_{c=1}^{Q}\sum_{k=1}^{N}\left(\omega_{+k}^c + \omega_{-k}^c\right) \leq N & \\
& \sum_{k:h \preceq k}\sum_{c=1}^{Q}\left(\omega_{+k}^c + \omega_{-k}^c\right) \leq N\, z_h & \forall h = 1,2,\ldots,N \\
& y \text{ unrestricted} & \\
& \omega_{+k}^c \geq 0 & \forall k = 1,2,\ldots,N;\ c \in \mathcal{C} \\
& \omega_{-k}^c \geq 0 & \forall k = 1,2,\ldots,N;\ c \in \mathcal{C} \\
& \beta^c \geq 0 & \forall c \in \mathcal{C} \\
& z_k \in \{0,1\} & \forall k = 1,2,\ldots,N.
\end{array}
\tag{5.9}
$$

We focus on the generation of Pareto-optimal solutions of Problem (5.7) for a scaled $L_1$-norm by using formulation (5.9) as dicussed below. The very

same approach can be used if one chooses any other polyhedral norm, such as the $L_\infty$ norm, instead of the $L_1$ norm, in Definition 1.2.

Problem (5.9) is a biobjective mixed integer linear problem, which can be tackled for instance, by adapting the two-phase method of [70] designed for solving biobjective knapsack problems.

In the first phase, one obtains the so-called supported solutions, namely, those which are found as solution of the scalarized problem

$$\begin{array}{ll} \max & \lambda_1\theta(W,b) - \lambda_2\pi(W,b) \\ \text{s.t.:} & \|W\|^\circ \leq 1 \\ & (W,b) \in I\!R^{NQ} \times I\!R^Q_+ \end{array} \tag{5.10}$$

for some weights $\lambda_1, \lambda_2 \in [0,1]$, with $\lambda_1 + \lambda_2 = 1$. These points describe, in the outcome space, the frontier of the convex hull of the Pareto-optimal outcomes.

Since we face a biobjective problem, the set of possible weights

$$\Lambda = \{(\lambda_1, \lambda_2) \in I\!R^2_+ : \lambda_1 + \lambda_2 = 1\}$$

that describe the supported efficient outcomes is unidimensional, and only a finite number of weights describe different outcomes. This fact can be exploited to find all supported outcomes in a sequential way.

A solution with minimal (zero) cost is the trivial solution $(W,b) = (0,0)$. Note that with this solution, points are classified arbitrarily by the tie-break rules, since all the components of the score functions will be zero.

When we are optimizing only the first objective, namely maximizing the margin, the optimal value can be obtained by solving Problem (5.5), which can be easily reformulated as a Linear Program. Denote by $\theta^*$ its optimal value. Given an optimal solution $(W^*, b^*)$ of Problem (5.5), a feasible solution $(W^*, b^*, z^*)$ of the biobjective problem (5.7) can be built by setting

$$z_i^* = \begin{cases} 1, & \text{if } \omega_i^{*c} \neq 0 \text{ for some } c \in \mathcal{C}, \\ 0, & \text{otherwise.} \end{cases}$$

If $(W^*, b^*)$ is the unique optimal solution, then $(W^*, b^*, z^*)$ will be a Pareto-optimal point. Otherwise, a Pareto-optimal point of Problem (5.7) can be found by maximizing the margin, i.e., by solving,

$$\begin{array}{ll} \min & \pi(W,b) \\ \text{s.t.:} & \|W\|^\circ \leq 1 \\ & \theta(W,b) \geq \theta^* \\ & (W,b) \in I\!R^{NQ} \times I\!R^Q_+. \end{array}$$

Once we have both a Pareto-optimal solution with minimal cost, i.e. $(0,0)$, and a Pareto-optimal solution with maximal margin, namely $(W_0, b_0)$, we construct an ordered list (sorted by either margin or by cost) whose elements can be built from any two consecutive already known elements $(W_1, b_1)$ and $(W_2, b_2)$ by the scalarized Problem (5.10) for certain $\lambda_1$ and $\lambda_2$. Denote $\theta^1$ and $\theta^2$ the margin of solution $(W_1, b_1)$ and $(W_2, b_2)$ respectively and costs $\pi^1$ and $\pi^2$. The scalarization needed in the problem is

$$\lambda_1 = \frac{\theta^2 - \theta^1}{\theta^2 - \theta^1 + \pi^2 - \pi^1}$$

$$\lambda_2 = \frac{\pi^2 - \pi^1}{\theta^2 - \theta^1 + \pi^2 - \pi^1}.$$

All optimal solutions of such scalarized problem are Pareto-optimal points. If both (or any of) $(W_1, b_1)$ and $(W_2, b_2)$ are solutions of the scalarized problem, the set of its optimal solutions yield the only supported Pareto outcomes between those of $(W_1, b_1)$ and $(W_2, b_2)$, so we do not need to seek more supported Pareto points between them. Since the number of Pareto outcomes is finite, the process ends in finite time.

When all the supported Pareto outcomes are found, the non-supported ones may be obtained in the following way. Let $(W_1, b_1)$ be any Pareto-optimal point with cost $\pi^1 > 0$. Let $\hat{\pi}$ be the minimal feature cost that is positive,

$$\hat{\pi} = \min_{k=1,2,\ldots,N} \{\Pi_k : \Pi_k > 0\}.$$

Then a Pareto-optimal point, with cost strictly lower than $\pi^1$, is obtained by solving the problem

$$\begin{array}{ll} \max & \theta(W,b) \\ \text{s.t.:} & \|W\|^\circ \leq 1 \\ & \pi(W,b) \leq \pi^1 - \hat{\pi} \\ & (W,b) \in I\!R^{NQ} \times I\!R^Q_+. \end{array} \tag{5.11}$$

Then, the next Pareto-optimal point can be found in the same way. Thus, starting from any supported Pareto-optimal point with cost greater than zero, the non-supported Pareto-optimal outcomes between it and the next supported one can be found.

## 5.6 Soft-margin biobjective optimization

In order to both avoid overfitting and deal with the non-separability of $I$, we now derive, as in Section 1.3.2, a soft-margin version of biobjective Problem (5.8), based on allowing some objects in $I$ to be misclassified. This is done, as in Problem (1.19), by adding to the model some slack variables $\eta^u \in \mathbb{R}^n_+$, where $n$ is the cardinal of the training sample. Using this idea, the biobjective Problem (5.8) is replaced by the following problem:

$$
\begin{array}{lll}
\max & y & \\
\min & \sum_{k=1}^{N} \Pi_k z_k & \\
\text{s.t.:} & \sum_{k=1}^{N} \phi_k(x^u)\left(\omega_k^i - \omega_k^j\right) + \beta^i - \beta^j - y + \eta^u \geq 0, & \\
 & & \forall i \neq j;\ i,j \in \mathcal{C},\ u \in I_i \\
 & \|W\|^\circ + C \sum_{u \in I} \eta^u \leq N & \\
 & -N z_k \leq \sum_{k:h \preceq k} \sum_{c=1}^{Q} \omega_k^c \leq N z_h & \forall h = 1,2,\ldots,N \\
 & \omega_k^c \text{ unrestricted} & \forall k = 1,2,\ldots,N;\ c \in \mathcal{C} \\
 & y \text{ unrestricted} & \\
 & \beta^c \geq 0 & \forall c \in \mathcal{C} \\
 & z_k \in \{0,1\} & \forall k = 1,2,\ldots,N \\
 & \eta^u \geq 0 & \forall u \in I,
\end{array}
\tag{5.12}
$$

for some user-defined value $C$, which trades off the perturbations $\eta^u$ and the margin.

In the same way as for the hard-margin approach, when $\|\cdot\|^\circ$ is a polyhedral norm, this problem can be formulated as a Biobjective Mixed Integer Problem. For instance, if $\|\cdot\|^\circ$ is a scaled $L_1$-norm, then Problem (5.12) can

be formulated as follows:

$$
\begin{array}{lll}
\max & y & \\
\min & \sum_{k=1}^{N} \Pi_k z_k & \\
\text{s.t.:} & \sum_{k=1}^{N} \phi_k(x^u)\left(\omega_{+k}^i - \omega_{-k}^i - \omega_{+k}^j + \omega_{-k}^j\right) + \beta^i - \beta^j - y + \eta^u \geq 0, & \\
 & & \forall i \neq j;\ i,j \in \mathcal{C},\ u \in I_i \\
 & \sum_{c=1}^{Q} \sum_{k=1}^{N} \left(\omega_{+k}^c + \omega_{-k}^c\right) + C \sum_{u \in I} \eta^u \leq N & \\
 & \sum_{k:h \leq k} \sum_{c=1}^{Q} \left(\omega_{+k}^c + \omega_{-k}^c\right) \leq N\, z_h & \forall h = 1,2,\ldots,N \\
 & y \text{ unrestricted} & \\
 & \omega_{+k}^c \geq 0 & \forall k = 1,2,\ldots,N;\ c \in \mathcal{C} \\
 & \omega_{-k}^c \geq 0 & \forall k = 1,2,\ldots,N;\ c \in \mathcal{C} \\
 & \beta^c \geq 0 & \forall c \in \mathcal{C} \\
 & z_k \in \{0,1\} & \forall k = 1,2,\ldots,N \\
 & \eta^u \geq 0 & \forall u \in I
\end{array}
\tag{5.13}
$$

The Two-Phase Method proposed in Section 5.5 to find the Pareto-optimal classifiers can also be used for solving Problem (5.13). Note that in this case, the solution with minimal (zero) cost is not the trivial solution $(W, b) = (0, 0)$, but any optimal solution of Problem (5.13) with $W$ set equal to the null matrix. The following steps of the method remain analogous to the hard-margin approach, and will not be repeated here.

## 5.7 Numerical results

In order to explore both, costs and quality, of the Pareto score functions obtained, we have performed a series of numerical tests on six standard databases, publicly available from the UCI Machine Learning Repository [6]: `bupa`, `credit`, `pima`, `thyroid`, `vehicle` and `wdbc`. A brief description of them is presented in Appendix A. For the sake of simplicity, the features are chosen as the original variables in the database $x_1, x_2, \ldots, x_p$ and their products, yielding monomials of degree up to $g$. However, other feature spaces, as those proposed by [7], might give better classification rates. For the `credit` database, only the eight quantitative variables where included.

Two types of costs are considered for the original variables. For the six databases, costs are independently chosen, randomly in the interval $(0, 1)$. Moreover, for the databases `bupa` and `pima` there exists a file, donated by Turney [66] and publicly available in the UCI repository [6], which contains an example for possible costs for the measurement of the variables. The cost

information comes from the Ontario Health Insurance Program's fee schedule. For these databases we have also considered such given costs. The remaining features have zero cost. The partial order is given as follows: feature $\phi = x_k$ precedes all features of the form $\phi(x) = x_k q(x)$ for some monomial $q(x)$ of degree up to $g - 1$.

Data were standardized by subtracting its mean and dividing by its standard deviation. Then, from each database, a random sample with two thirds of the objects is drawn and used as training sample $I$. The supported Pareto-optimal solutions of Problem (5.12) were computed by the first phase of the Two-Phase Method [70], described in Section 5.5. The non-supported Pareto-optimal solutions can also be computed using formulation (5.11). The trade-off parameter $C$ is chosen to be equal to the number of objects in $I$.

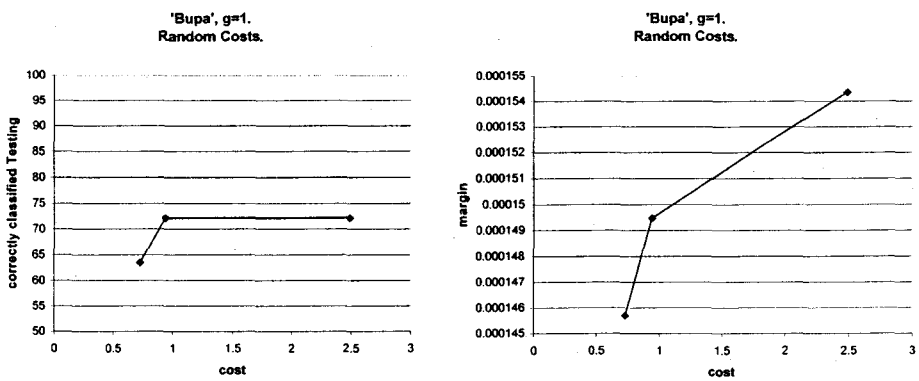


*Fig. 5.1:* Database 'bupa', $g = 1$, random costs

The results are plotted in Figures 5.1-5.10. In the right side of such figures, measurement costs of the Pareto-optimal rules (except for zero-cost solutions) are plotted against the margin. Since only Pareto-optimal solutions are considered, we see that, the higher the cost, the higher the margin.

This is the plot the final user will obtain in real-world applications, and choose, with this information, one classification rule.

However, margin maximization is only a surrogate for the minimization of the misclassification rate, which will remain unknown. In the left side of Figures 5.1-5.10 we have plotted, for the Pareto-optimal classifiers obtained, costs against the percentage of correctly classified objects in the testing sample. Figures show clearly that high correct classification rates correspond to

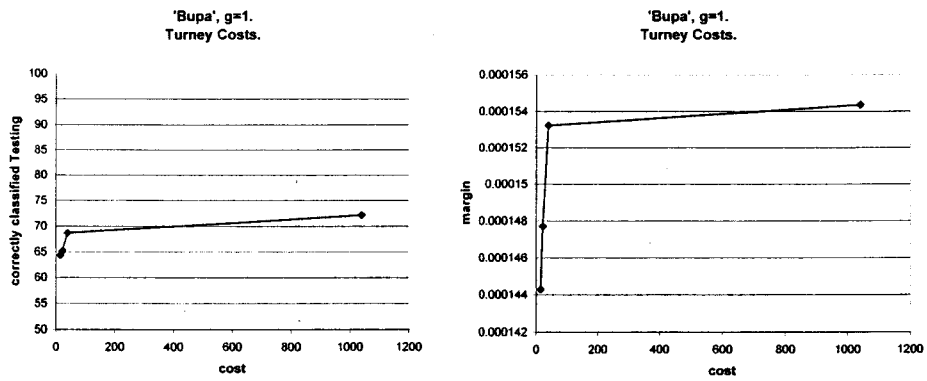


*Fig. 5.2:* Database 'bupa', $g = 1$, Turney's costs

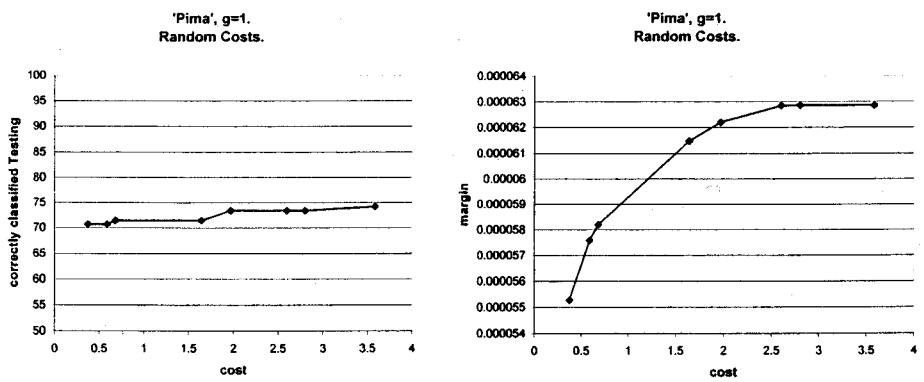


*Fig. 5.3:* Database 'pima', $g = 1$, random costs

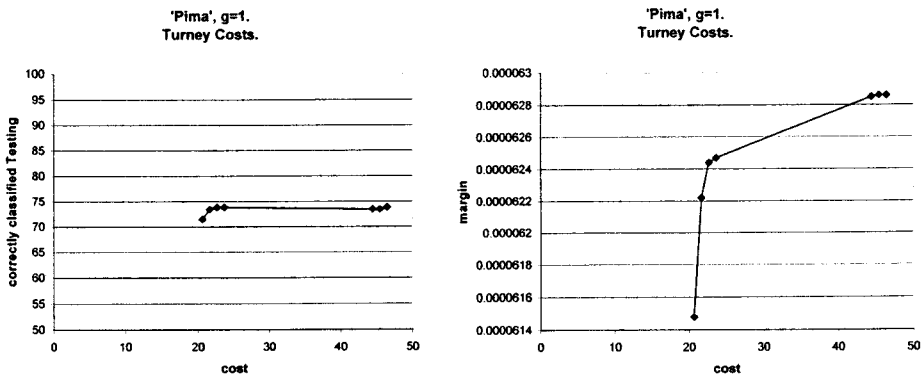


*Fig. 5.4:* Database 'pima', $g = 1$, Turney's costs

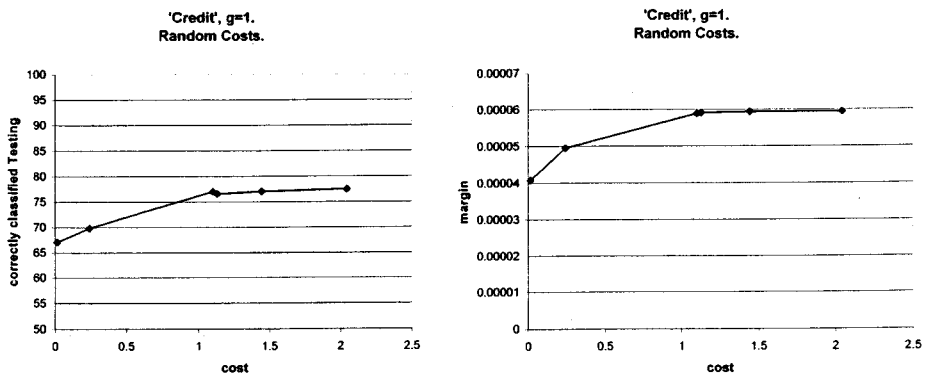


*Fig. 5.5:* Database 'credit', $g = 1$, random costs

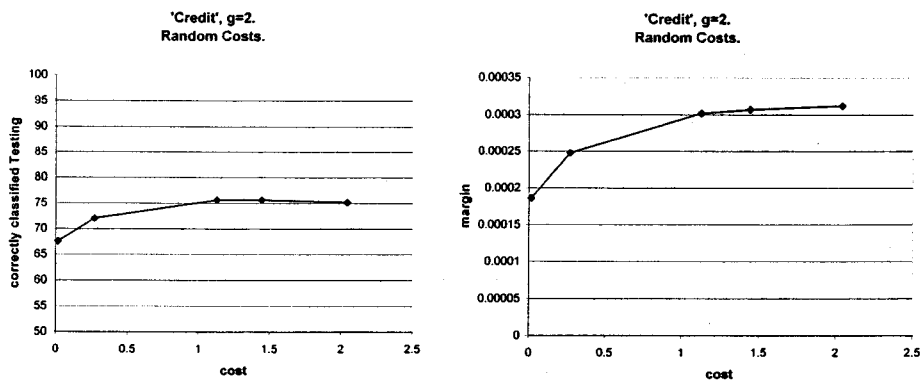


*Fig. 5.6:* Database 'credit', $g = 2$, random costs

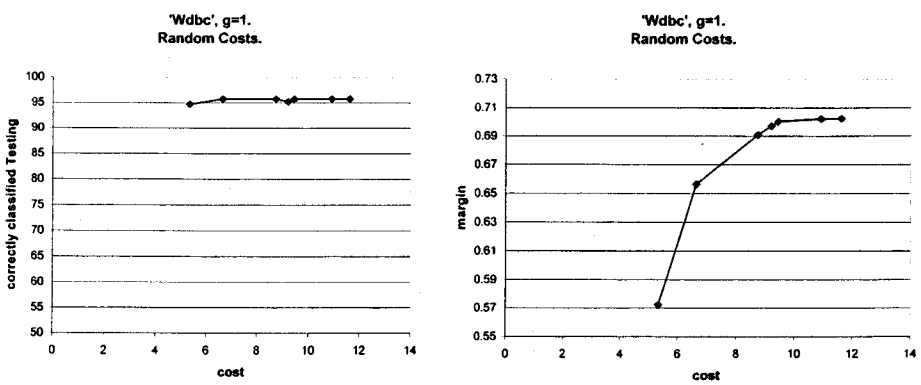


*Fig. 5.7:* Database 'wdbc', $g = 1$, random costs

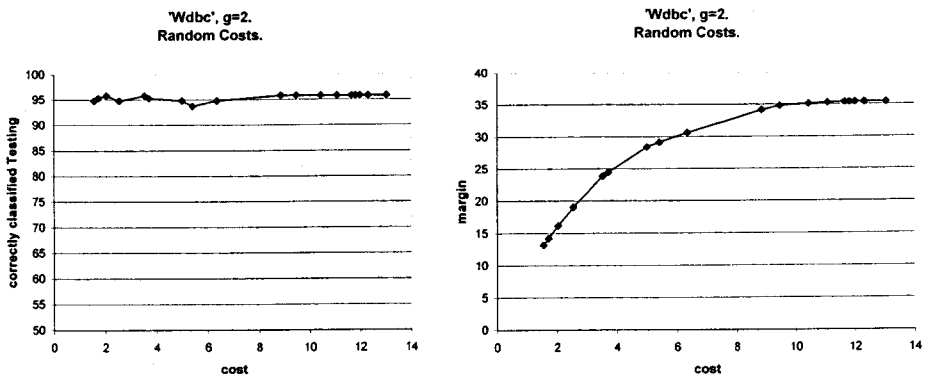


*Fig. 5.8:* Database 'wdbc', $g = 2$, random costs

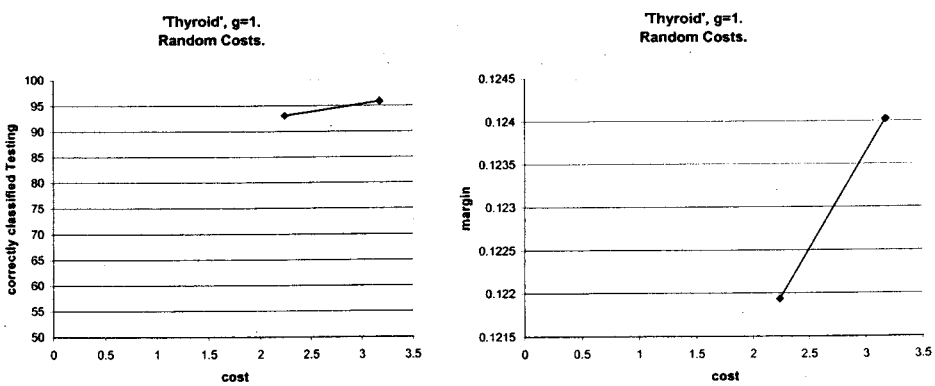


*Fig. 5.9:* Database 'thyroid', $g = 1$, random costs

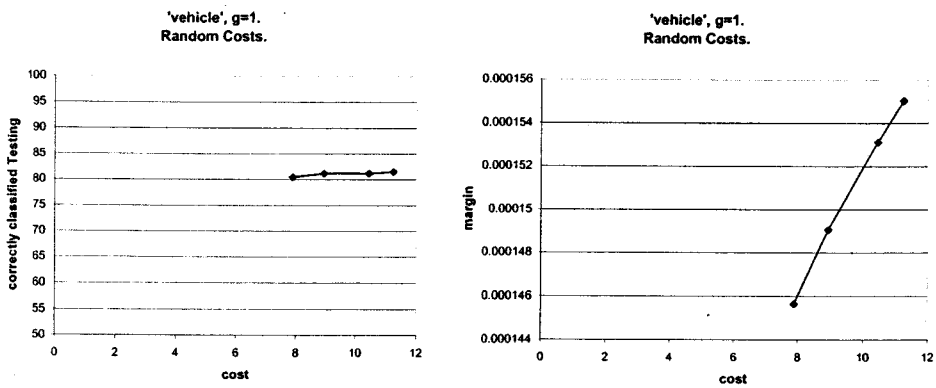


*Fig. 5.10:* Database 'vehicle', $g = 2$, random costs

high costs. Moreover, the trade-off between measurement costs and margin translates into a similar trade-off between measurement costs and percentage of correctly classified objects.

| method | bupa | credit | pima | thyroid | vehicle | wdbc |
|---|---|---|---|---|---|---|
| 1-Nearest Neighbor | 60.87 | 72.07 | 64.84 | 97.22 | 68.09 | 94.74 |
| 2-Nearest Neighbor | 57.39 | 70.72 | 69.14 | 94.44 | 68.09 | 94.21 |
| 3-Nearest Neighbor | 60.00 | 73.87 | 72.27 | 94.44 | 70.57 | 95.26 |
| 4-Nearest Neighbor | 60.87 | 72.52 | 72.27 | 93.06 | 71.28 | 95.26 |
| 5-Nearest Neighbor | 62.61 | 72.07 | 71.48 | 93.06 | 71.28 | 95.79 |
| Classification Tree | 67.83 | 72.97 | 70.31 | 93.06 | 66.67 | 90.53 |
| linear SVM | 72.17 | 77.48 | 74.22 | 95.83 | 81.21 | 95.79 |
| polynomial SVM, $g = 2$ | 66.96 | 65.32 | 38.28 | 94.44 | 78.37 | 94.21 |
| polynomial SVM, $g = 3$ | 59.13 | 69.37 | 66.41 | 91.67 | 79.79 | 93.68 |
| polynomial SVM, $g = 4$ | 58.26 | 59.01 | 62.89 | 93.06 | 76.24 | 89.47 |
| polynomial SVM, $g = 5$ | 57.39 | 75.23 | 67.19 | 94.44 | 74.11 | 91.58 |
| rbf SVM | 68.70 | 77.48 | 64.84 | 88.89 | 66.67 | 63.16 |

*Tab. 5.2:* Behavior of other methods

For comparative purposes, in Table 5.2, the percentage of correctly classified objects is shown for different classification methods, such as classification trees [11], $k-$nearest neighbor classifier [22] and the classical SVM approach as implemented in SVMlight [44]. For the databases with more than two classes, mainly `thyroid` and `vehicle`, the one-against-one approach was

used, as implemented in LIBSVM [20]. It can be observed that the classification behavior of the Pareto-optimal classifiers are among the best ones, even for low classification costs.

## 5.8 Conclusions

In this chapter, we propose, for multigroup classification, a model that takes into account measurement costs for the predictor variables. It is formulated as a Biobjective Mixed Integer Program, for which a Two-Phase method is proposed in order to find its Pareto-optimal solutions. Among such solutions, it is up to the practitioner choosing one of them, taking into account the margin, surrogate of the classification quality, and the cost. Our numerical examples show, for the supported Pareto-optimal solutions, the trade-off between cost and margin, and also between cost and the classification quality in a testing sample.

The method proposed in this chapter, can thus be seen as a procedure that generates a series of classification rules with different costs, and expected good classification behavior supported by the theoretical generalization properties of the margin maximizer (e.g. Vapnik [68]). Choosing one classification rule among them can be done by the practitioner after plotting the measurement costs against margins, as illustrated in the examples. To have a better idea about the classification quality of the rules in new objects, the practitioner could use a testing sample.

The choice of the $L_1$ norm allowed us to formulate the model as a Biobjective Mixed Integer Program, solved after addapting an already existing tool: the Two-Phase method. Methods to find the Pareto-optimal solutions when other norms, like the Euclidean, are of great interest, specially for its ability to include other kinds of embedding or kernels.

# 6. FINAL REMARKS

The classification problem is one of the main tasks in Data Mining. In this thesis we have shown the great potential of Mathematical Programming tools, and in particular Linear Programming, to model and solve classification problems where good classification quality is not the only aim, but other properties are desirable for the obtained classifier.

When SVM is solved as a Linear Program, advanced techniques such as Column Generation can be used to solve problems that otherwise will be difficult to manage. For example, when all the cutoffs and all the interactions between pair of variables are taken into account, the Linear Program is too large to be solved by standard techniques. Moreover, in the Column Generation algorithms proposed in Chapters 2 and 3, columns correspond to features (defined by a variable with its cutoff, or a set of variables interacting). Thus, the pricing problem that generate the columns actually generates such features. The study of pricing problems that include statistical information about the variables may be of interest and could accelerate the generation process. For instance, we could use, measures used in Classification Trees to decide the cutoff and variables for the split (like entropy, information gain, Gini impurity), can be used alone, or together with the pricing problem, to generate new columns or features.

We could also add constraints in the features we allow to be used. For instance, in [7] they propose to use a set of features (they called them positive and negative patterns) which are similar to the presented in Chapter 3, but imposing certain properties on how, when they are not combined with the others, they classify objects in $I$. They then propose several ways to combine these patterns, one of them is margin maximization. However, in their model all the patterns should be at hand, whereas, by extending the column generation model proposed in this thesis, patterns could be generated as needed.

In Chapter 4, we formulate the problem of simultaneous maximization of the margin on each class, defined independently to the other class. This prob-

lem is useful to find classifiers when misclassifying an object is not equally important depending on the class the object belongs to. Multicriteria optimization allow us to analyze this situation and characterize the classifiers that can not be improved simultaneously in both classes. In the Euclidean case, this result provides a new theoretical foundation for the use of the widely used tool called ROC curve.

In addition, Linear Programming and Mixed Integer Programming tools for exactly solving multicriteria problems have been widely developed, allowing us to find a reduced set of good classifiers, in the sense that they can not be improved simultaneously in all of the desired properties. The practitioner has the opportunity of picking up one of them by trading off the different aims. An example of this situation is presented in Chapter 5, with a model in which cheap classifiers with good classification quality are sought.

# APPENDIX

# A. DATABASES DESCRIPTION

In this section we briefly describe the different databases, all of them are publicly available at the UCI Machine Learning Repository [6], used in the computational experiments and numerical examples, namely,

- the Cylinder Bands Database, called here `bands`;
- the BUPA Liver-disorders Database, called here `bupa`;
- the Credit Screening Databases, called here `credit`;
- the Ionosphere Database, called here `ionosphere`;
- the Pima Indians Diabetes Database, called here `pima`;
- the Sonar Database, called here `sonar`;
- the Thyroid Disease Database, called here `thyroid`;
- the Vehicle Silhouettes Database, called here `vehicle`;
- and the New Diagnostic Database contained in the Wisconsin Breast Cancer Databases, called here `wdbc`.

For each database, the name of the file (as called in the database), the total number of objects $|\Omega|$, the number of groups $Q$, the number of variables (all quantitative, after the preprocessing explained in Section 2.2) $p$ and the type of data are given in Table A.1. Under the column `type`, `q` denotes that the database contains only numeric variables and `m` that it is a mixed database with numeric, binary and qualitative variables.

In case of existence of missing values, as occurs in `bands`, `credit`, objects with missing values have been removed from the database.

| Database | filename | $\|\Omega\|$ | $Q$ | $p$ | type |
|---|---|---|---|---|---|
| bands | bands.data | 365 | 2 | 56 | m |
| bupa | bupa.data | 345 | 2 | 5 | q |
| credit | crx.data | 666 | 2 | 43 | m |
| ionosphere | ionosphere.data | 351 | 2 | 34 | q |
| pima | pima-indians-diabetes.data | 768 | 2 | 8 | q |
| sonar | sonar.all-data | 208 | 2 | 60 | q |
| thyroid | new-thyroid | 215 | 3 | 5 | q |
| vehicle | vehicle.data | 846 | 4 | 18 | q |
| wdbc | wdbc.data | 569 | 2 | 30 | q |

*Tab. A.1:* Description of the databases

# BIBLIOGRAPHY


[1] S. Alexe, E. Blackstone, P. Hammer, H. Ishwaran, M. Lauer, and C.E. Pothier Snader. Coronary risk prediction by logical analysis of data. *Annals of Operations Research*, 119:15–42, 2003.

[2] C. Apte. The big (data) dig. *OR/MS Today*, February 2003.

[3] C. Apte, B. Liu, E.P.D. Pednault, and P. Smyth. Business applications of data mining. *Communications of the ACM*, 45:49–53, 2002.

[4] V. Bayer-Zubek. *Learning Cost-Sensitive Diagnostic Policies from Data.* PhD thesis, Oregon State University, July 2003. `http://eecs.oregonstate.edu/library/?call=2003-13`.

[5] K.P. Bennet and O.L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–24, 1992.

[6] C.L. Blake and C.J. Merz. UCI Repository of Machine Learning Databases. `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 1998. University of California, Irvine, Dept. of Information and Computer Sciences.

[7] E. Boros, P. L. Hammer, T. Ibaraki, and A. Kogan. A logical analysis of numerical data. *Mathematical Programming*, 79:163–190, 1997.

[8] P.S. Bradley, U.M. Fayyad, and O.L. Mangasarian. Mathematical programming for data mining: formulations and challenges. *INFORMS Journal on Computing*, 11(3):217–238, 1999.

[9] P.S. Bradley, O. Mangasarian, and D. Musicant. Optimization methods in massive datasets. In J. Abello, P.M. Pardalos, and M.G.C. Resende, editors, *Handbook of Massive Datasets*, pages 439–472. Kluwer Academic Publishers, 2002.

[10] E. Bredensteiner and K. Bennet. Multicategory classification by support vector machines. *Computational Optimization and Applications*, 12:53–79, 1999.

[11] L. Breiman, J.H. Friedmann, R.A. Olshen, and C.J. Stone. *Classification and Regresion Trees.* Wadsworth, 1984.

[12] E. Carrizosa and J. Fliege. Generalized goal programming: Polynomial methods and applications. *Mathematical Programming*, 93:281–303, 2002.

[13] E. Carrizosa and B. Martín-Barragán. Problemas de clasificación: una mirada desde la localización. In Blas Pelegrín, editor, *Avances en localización de servicios y sus aplicaciones.* Servicio de Publicaciones de la Universidad de Murcia, 2005.

[14] E. Carrizosa and B. Martín-Barragán. Two-group classification via a biobjective margin maximization model. *European Journal of Operational Research*, 2006. In press.

[15] E. Carrizosa, B. Martín-Barragán, and D. Romero Morales. A biobjective model to select features with good classification quality and low cost. In *Proceedings of the Fourth IEEE ICML*, pages 339–342. IEEE Publications, 2004.

[16] E. Carrizosa, B. Martín-Barragán, and D. Romero Morales. A column generation approach for support vector machines. Technical report, Optimization Online, 2006. `http://www.optimization-online.org/DB_HTML/2006/04/1359.html`.

[17] E. Carrizosa, B. Martín-Barragán, and D. Romero Morales. Detecting relevant variables and interactions for classification in support vector machines. Technical report, Optimization Online, 2006. `http://www.optimization-online.org/DB_HTML/2006/05/1385.html`.

[18] E. Carrizosa, B. Martín-Barragán, F. Plastria, and D. Romero Morales. On the selection of the globally optimal prototype subset for Nearest Neighbor classification. *To appear in INFORMS Journal on Computing*, 2006.

[19] E. Carrizosa and F Plastria. Optimal expected-distance separating half-space. Technical Report MOSI/7, Vrije Universiteit Brussel, 2004.

[20] C.C. Chang and C.J. Lin. LIBSVM: A library for support vector machines. (Downloadable from website `http://www.csie.ntu.edu.tw/~cjlin/libsvm`), 2001.

[21] C. Cortes and V. Vapnik. Support-vector network. *Machine Learning*, 1:113–141, 1995.

[22] T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[23] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines.* Cambridge University Press, 2000.

[24] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1):225–254, 2002.

[25] A.P. Duarte Silva and A. Stam. Second order mathematical programming formulations for discriminant analysis. *European Journal of Operational Research*, 72:4–22, 1994.

[26] B. Efron and R. Tibshirani. Improvements on cross-validation: The .632+ bootstrap method. *Journal of the American Statistical Association*, 92(438):548–560, 1997.

[27] B. Efron, R. Tibshirani, J. Storey, and V. Tusher. Empirical bayes analysis of a microarray experiment. *Journal of the American Statistical Association*, 96:1151–1160, 2001.

[28] M. Ehrgott and X. Gandibleaux, editors. *Multiple Criteria Optimization. State of the Art Annotated Bibliographic Surveys*, volume 52 of *International Series in Operations Research and Management Science.* Kluwer Academic Publishers, Boston, 2002.

[29] J.E. Falk and V.E. Karlov. Robust separation of finite sets via quadratics. *Computers and Operations Research*, 28:537–561, 2001.

[30] U. Fayyad and R. Uthurusamy. Evolving data mining into solutions for insight. *Communications of the ACM*, 45:28–31, 2002.

[31] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[32] P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting-stock problem. *Operations Research*, 9:849–859, 1961.

[33] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.

[34] Y. Guermeur. Combining discriminant models with multi-class SVMs. *Pattern Analysis and Applications*, 5:168–179, 2002.

[35] I. Guyon and A. Elisseff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3(1157-1182), 2003.

[36] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[37] J. Han, R.B. Altman, V. Kumar, H. Mannila, and D. Pregibon. Emerging scientific applications in data mining. *Communications of the ACM*, 45:54-58, 2002.

[38] H. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining.* The MIT Press, 2001.

[39] T. Hastie and R. Tibshirani. Classification by pairwise coupling. *The Annals of Statistics*, 26(2):451–471, 1998.

[40] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, 2001.

[41] R. Herbrich. *Learning Theory Classifiers. Theory and Algorithms.* MIT Press, 2002.

[42] ILOG CPLEX 8.1 User's Manual. `http://www.pcs.cnu.edu/~riedl/software/cplex81/doc/userman/onlinedoc`.

[43] Informe de intel sobre la ley de moore. `http://www.intel.com/technology/mooreslaw/index.htm`.

[44] T. Joachims. *Learning to Classify Text Using Support Vector Machines.* Kluwer, 2002.

[45] KDnuggets. `http://www.kdnuggets.com/about/index.html`.

[46] KDnuggets : Polls : Important data mining topics. `http://www.kdnuggets.com/polls/2005/important_data_mining_topics.htm`.

[47] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 1137–1143. Morgan Kaufmann, 1995.

[48] M.A. Kupinski and M.A. Anastasio. Multiobjective genetic optimization of diagnostic classifiers with implications for generating receiver operating characteristic curves. *IEEE Transactions On Medical Imaging*, 18(8):675–685, 1999.

[49] O.L. Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, 13:444–452, 1965.

[50] O.L. Mangasarian. Mathematical programming in data mining. *Data Mining and Knowledge Discovery*, 42(1):183–201, 1997.

[51] O.L. Mangasarian. Arbitrary-norm separating plane. *Operations Research Letters*, 24:15–23, 1999.

[52] H. Martini and A. Schöbel. Median and center hyperplanes in minkowski spaces –a unifying approach. *Discrete Mathematics*, 241:407–426, 2001.

[53] S.W. Norton. Generating better decision trees. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, IJCAI-89*, pages 800–805, Detroit, Michigan, 1989.

[54] M. Núñez. The use of background knowledge in decision tree induction. *Machine Learning*, 6:231–250, 1991.

[55] P. Paclík, R.P.W. Duin, G.M.P. van Kempen, and R. Kohlus. On feature selection with measurement cost and grouped features. *Lecture Notes in Computer Science*, 2396(461-469), 2002.

[56] J.P. Pedroso and N. Murata. Support vector machines with different norms: motivation, formulations and results. *Pattern recognition letters*, 22:1263–1272, 2001.

[57] S. Piramuthu. Evaluating feature selection methods for learning in data mining applications. *European Journal of Operational Research*, 156:483–494, 2004.

[58] F. Plastria and E. Carrizosa. Gauge distances and median hyperplanes. *Journal of Optimization Theory and Applications*, 110:173–182, 2001.

[59] F. Plastria and E. Carrizosa. Optimal distance separating halfspace. Technical report, BEIF/124, Vrije Universiteit Brussel, 2002.

[60] J.C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. *Advances in Neural Information Processing Systems*, 12:547–553, 2000.

[61] A.M. Rubinov, A.M. Bagirovand, N.V. Soukhoroukova, and J. Yearwood. Unsupervised and supervised data classification via nonsmooth and global optimization. *TOP*, 11(1):1–93, 2003.

[62] A. Schwaighofer. SVM toolbox for Matlab. (Downloadable from website `http://www.cis.tugraz.at/igi/aschwaig/software.html`), 2002.

[63] D.K. Slonim. From patterns to pathways: gene expression data analysis comes of age. *Nature Genetics Supplement*, 32:502–508, 2002.

[64] A. Stam. Nontraditional approaches to statistical classification: Some perspectives on $l_p$-norm methods. *Annals of Operations Research*, 74:1–36, 1997.

[65] M. Tan. Cost-sensitive learning of classification knowledge and its applications in robotics. *Machine Learning*, 13:7–33, 1993.

[66] P.D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409, 1995.

[67] V. Vapnik. *The nature of statistical learning theory.* Springer-Verlag, 1995.

[68] V. Vapnik. *Statistical learning theory.* Wiley, 1998.

[69] K. Veropoulos, N. Cristianini, and C. Campbell. Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on Artificial Intelligence, (IJCAI99)*, pages 55–60, 1999.

[70] M. Visée, J. Teghem, M. Pirlot, and E.L. Ulungu. Two-phases method and branch and bound procedures to solve the biobjective knapsack problem. *Journal of Global Optimization*, 12:139–155, 1998.

[71] G. Wahba, Y. Lin, and H. Zhang. Generalized approximate cross validation for support vector machines. In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 297–311. MIT Press, 2000.

[72] S.M. Weiss and C.A. Kulikowski. *Computer Systems That Learn.* Morgan Kaufmann, 1999.

[73] Weka 3: Data mining software in java. `http://www.cs.waikato.ac.nz/ml/weka/`.

[74] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 668–674. MIT Press, 2001.

[75] J. Weston and Watkins. Multi-class support vector machines. In *Proceedings of ESANN99, Brussels, D. Facto Press*, 1999.

[76] I.H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann, 2005.

[77] D. Xie, S.B. Singh, E.M. Fluder, and T. Schlick. Principal component analysis combined with truncated-Newton minimization for dimensionality reduction of chemical databases. *Mathematical Programming*, 95(1):161–185, 2003.

[78] C. Zopounidis and M. Doumpos. Multicriteria classification and sorting methods. *European Journal of Operational Research*, 138:229–246, 2002.

## UNIVERSIDAD DE SEVILLA

Reunido el tribunal en el día de la fecha, integrado por los abajo firmantes, para evaluar la tesis doctoral de Dª BELÉN MARTÍN BARRAGÁN titulada PROGRAMACIÓN MATEMÁTICA PARA LAS MÁQUINAS DE VECTOR DE APOYO

acordó otorgarle la calificación de SOBRESALIENTE CUM LAUDE POR UNANIMIDAD

Sevilla, a 20 de SEPTIEMBRE de 2006.

Vocal,

Vocal,

Vocal,

Presidente,

Secretario,

Doctorando,