Universidad Carlos III de Madrid

Doctoral Dissertation

Support Vector Machine Tools for Multi-class Classification Problems

Author: **Ling Liu**

Advisors:
Francisco Javier Prieto
Universidad Carlos III de Madrid
Belen Martin-Barrangan
University of Edinburgh, Business School

Department of Statistics
Madrid, March 2016

Universidad
Carlos III de Madrid
www.uc3m.es

TESIS DOCTORAL

Support Vector Machine Tools for Multi-class Classification Problems

Autor:

Ling Liu

Director/es:

Belén Martín Barragán

Tutor:

Francisco Javier Prieto Fernández

DEPARTAMENTO DE ESTADÍSTICA

Leganés, marzo, 2016

Universidad Carlos III de Madrid
www.uc3m.es

TESIS DOCTORAL

Support Vector Machine Tools for Multi-class Classification Problems

Autor: *Ling Liu*

Director/es: **Francisco Javier Prieto Fernández, Belén Martín Barragán**

Firma del Tribunal Calificador:

Firma

Presidente: Emilio Jesús Carrizosa Priego

Vocal: David Ríos Insua

Secretario: Francisco Javier Nogales Martín

Calificación:

Leganés, de de

To my family

Acknowledgements

First of all, I would like to express my sincere gratitude to my advisors Francisco Javier Prieto and Belen Martin-Barrangan for the continuous support of my Ph.D study and related research, for their patience, motivation, encouragements, trust and immense knowledge. Their guidance helped me a lot in all the time of research and writing of this thesis. I will always be thankful for their great help.

I am glad that I have the chance to spend the last five years in the Statistic department of Universidad Carlos III de Madrid which is located in a such sunshine and warm city.

My sincere thanks also goes to my good friends and college mates. Especially to Xiaoling Mei and Xiuping Mao for always being my best friends. Both of you have helped me a lot when I was depressed. I will always remember the last ten years that we have spent together. And I would also like to thank my roommates for giving me the attentive care of my sick time and also the happy weekends.

Finally, I would like to give my deepest gratitude to my parents and my boyfriend Xie Fei for their unconditional supporting and understanding.

Abstract

Lately, Support Vector Machine (SVM) methods have become a very popular technique in the machine learning field for classification problems. It was originally proposed for classifications of two classes. The effectiveness of this method has not only been shown in hundreds of experiments, but also been proved in theory. In our real life, we usually have more than two classes. Various multi-class models with a single objective have been proposed mostly based on two families of methods: an all-together approach and a combination of binary classifiers. However, most of these single-objective models consider neither the different costs of different misclassifications nor the users' preferences. To overcome these drawbacks, we have two approaches. A direct way is to give different weights to the penalties in the objective functions. The difficulty for this way is how to choose proper values for the weights. Alternatively, multi-objective approaches have been proposed. However, these multi-objective approaches need to solve a set of large Second-Order-Cone Programs (SOCPs) and gives us weakly Pareto-optimal solutions.

This thesis is comprised of two working papers on multi-class SVMs. We summarize the contributions of these two working papers as follows.

In the first article, we propose a multi-objective technique that we denominate Projected Multi-objective SVM (**PM**), which works in a higher dimensional space than the object space. For **PM**, we can characterize its Pareto-optimal solutions. And for classifications with large numbers of classes, **PM** significantly alleviates the computational bottlenecks. From our experimental results, and compared with the single-objective multi-class SVMs (based on an all-together method, one-against-all method and one-against-one method), **PM** obtains comparable values for the training classification accuracies, testing classification accuracies and training time, with the advantage of providing a wider set of options, each of them designed for different misclassification costs. Compared to other multi-objective methods, **PM** outperforms them in terms of the out-of-sample quality of the approximation of the Pareto frontier, with a considerable reduction of the computational burden.

In the second article, we focus on finding the appropriate values of the weight parameters for the single-objective multi-class SVM which considers all classes in one quadratic program (QP). We propose a partial parametric path algorithm (**PPPA**) taking advantage of the piecewise linearity of the optimal solutions of the weighted single-objective SVMs with respect to the trade-off parameter C. Compared to the traditional grid search method which needs repeatedly solving the QPs, using **PPPA** we need to solve only one QP and several linear equations. Thus we can save a lot of computation using **PPPA**. To systematically explore the different weights

for the misclassification costs, we combine the **PPPA** with a variable neighborhood search method. Our numerical experiments shows the efficiency and reliability of **PPPA**.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Data mining has become a crucial application area in modern science and industry due to the growing size of available databases. One of the main applications in this area is supervised classification: to obtain a model that predicts the value of one variable (class) based on the information from other variables. Several approaches have been proposed to solve this problem, such as K-NN, Decision trees, Naive-Bayes classification, Neural networks, SVM and so on, see for example [Corne et al., 2012]. In this thesis we focus on the SVM approach, which is well known to have high generalization ability compared to other classification algorithms.

The development of new efficient solution techniques has happened in parallel to the increase in size and complexity of classification problems. In [Cortes and Vapnik, 1995], they proposed the classical binary SVM algorithm to separate the input data into two different classes. During the last couple of decades, hundreds of applications and experiments have shown the high classification accuracy of SVM classifiers, e.g. [Carrizosa and Martin-Barragan, 2006, Guyon et al., 2002, Heisele et al., 2001, Tong and Koller, 2001]. There are two types of theoretical explanations of the superior performance of the SVM procedures. The first is represented by the theoretical justification of SVM in Vapnik's structure risk minimization approach [Vapnik, 1995, 1998]. The other theoretical explantation of SVM's good performance is given by Lin in 2002, who identified the asymptotic target function of the SVM formulation and associated it with the Bayes decision rule, [Lin, 2002].

1.1 Classical binary SVM

We briefly introduce the classic binary SVM, as the basic reference for further developments, and in particular for the proposals in this dissertation. As in [Cortes and Vapnik, 1995], we assume that we have a training set $\{x_i\}_{i=1}^n \subseteq \mathbb{R}^l$, of objects from two classes " $+$ " and " $-$ ". Let $I = \{1, 2, \cdots, n\} \subseteq \mathbb{R}$ to denote the indexes of the training objects. Let $y_i = +1$ if x_i belongs to " $+$ " class and -1 otherwise. The aim is to use the training set I to construct a classification rule that predicts the class membership of objects with high accuracies. To achieve this aim,

we need to find a discriminant hyperplane that separates the training set with high accuracy and maximum margin. The discriminant hyperplane is defined as: $\omega^T x + \beta = 0$. And the class membership of a new object is predicted using the decision function:

$$f_{\omega,\beta}(x) = sign(\omega^T x + \beta).$$

The ideal case is when we have all the class " + " objects above the discriminant hyperplane and all the class " − " objects below the discriminant hyperplane (or viceversa). When the available data fits this ideal case, we call the binary training data "linearly separable", as shown in Figure 1.1.

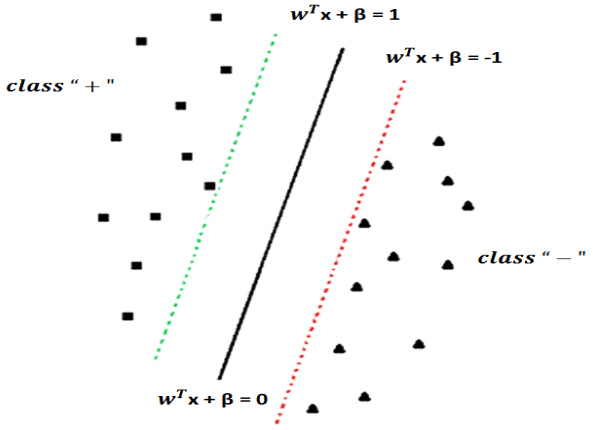


Figure 1.1: Binary linearly separable training data

For binary classification problems, when we have linearly separable data we introduce constraints to ensure that all the training instances should be correctly classified; the corresponding binary SVMs are called hard-margin binary SVMs. To avoid overfitting or if we have nonlinearly separable training data, we add slack to these constraints using auxiliary variables. These slack variables allow the presence of some misclassification errors; these classification errors are minimized in the objective function. The binary SVMs havin gthis structure are called soft-margin binary SVMs.

1.1.1 Hard-margin classical binary SVM

When all the training data can be linearly separated, a hard-margin classical binary SVM problem (1.1) is constructed to define the optimal hyperplane. This hyperplane is optimal not only because it can separate the training data correctly, but also because it presents the largest functional margin between the classes. The functional margin is calculated as the distance between the two supporting hyperplanes: $\omega^T x+\beta = 1$ and $\omega^T x+\beta = -1$. Note that maximizing

the functional margin $\sqrt{2}/\|\omega\|$ is equivalent to minimizing $\|\omega\|$.

$$\begin{aligned} \min_{\omega,\beta} \quad & \frac{1}{2}\|\omega\|^2, \\ \text{s.t.} \quad & y_i(\omega^T x_i + \beta) \geq 1, i \in I, \end{aligned} \tag{1.1}$$

where $\omega \in \mathbb{R}^l$ and $\beta \in \mathbb{R}$.

When the training data can be separated without errors, the expected value of the probability of misclassification can be bounded [Vapnik and Vapnik, 1982] as follows:

$$E[P(\text{error})] \leq \frac{E[\text{number of support vectors}]}{\text{number of training vectors}}.$$

This bound does not explicitly contain the dimensionality of the feature space, and as consequence, the generalization ability of the classification rule is not directly affected by this dimension, as opposed to the properties of other traditional statistical approaches.

1.1.2 Soft-margin classical binary SVM

For the nonlinearly separable case, the two main approaches are either mapping the data onto a higher dimensional space or allowing some objects in the training set to be misclassified. An example of the first approach is illustrated in Figure 1.2. For this example, in $\mathbb{R}^2$, the data can

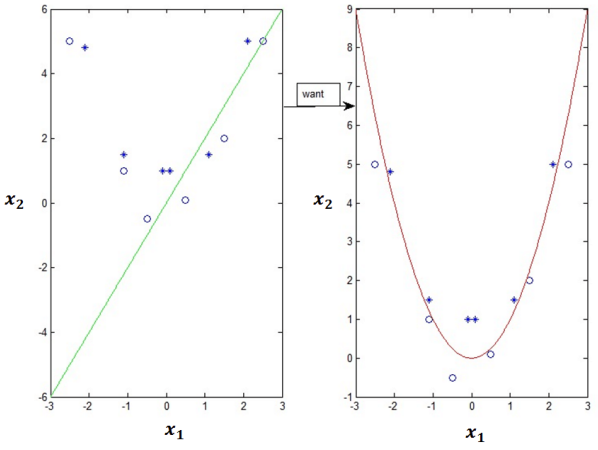


Figure 1.2: A nonlinearly separable example

not be linearly separated. With the help of the transformation $\phi((x_1, x_2)) = (x_1, x_1^2, x_2)$, we can linearly separate the (modified) data in $\mathbb{R}^3$.

The ideal case is that we can find a map function $\phi(x)$, with which we can separate the training data. However, it may be hard to find the proper $\phi(x)$. In practice, we usually accept the presence of some classification errors. Then, by maximizing the functional margin and minimizing

the classification errors, a soft-margin classical binary SVM is defined as:

$$\min_{\omega,\beta} \ \frac{1}{2}\|\omega\|^2 + C\sum_{i\in I}(1 - y_i(\omega^T\phi(x_i) + \beta))_+,$$

where $\omega \in \mathbb{R}^l$, $\beta \in \mathbb{R}$ and C quantifies the tradeoff between the efficiency and generalization properties of the solution.

For a map function ϕ, the Kernel function is defined as $K(x_i, x_j) = \phi(x_i)^T\phi(x_j)$. And the corresponding dual problem is:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i\in I}\alpha_i - \frac{1}{2}\sum_{i\in I}\sum_{j\in I}\alpha_i\alpha_j y_i y_j K(x_i, x_j), \\ \text{s.t.} \quad & \sum_{i\in I}\alpha_i y_i = 0, \\ & 0 \le \alpha_i \le C, i \in I. \end{aligned} \tag{1.2}$$

In the dual problem (1.2), all the information related to the nonlinear transformation that is required is $K(x_i, x_j)$. So instead of defining a map function ϕ with the desired separation property, the "kernel trick" directly defines a proper Kernel function in the dual problem. This approach is widely used, [Lin and Lin, 2003, Rüping, 2001, Zhang et al., 2006], as it simplifies significantly the formulation of the SVM problem.

1.2 Unbalanced binary SVM

The classical binary SVM does not consider the difference between different misclassification costs, nor any a priori information that might be available (such as skewed class distributions). However, the use of this type of information may be critical for the efficiency of the method. For example, in medical diagnosis it is known that the cost of misclassifying a healthy person as ill is quite different from the one of misclassifying an ill patient as healthy. A direct way to solve this problem is to assign different weights to the different penalties in the objective function, as done in [Akbani et al., 2004, Veropoulos et al., 1999]. Alternatively, bi-objective SVMs can be used for binary classifications, [Carrizosa and Martin-Barragan, 2006].

1.2.1 Single-objective unbalanced binary SVM

Differences in misclassification costs, or the presence of skewed class distributions, affect the performance of the SVMs. As in [Akbani et al., 2004, Veropoulos et al., 1999], different weights can be assigned to the different misclassification penalties. Then, the single-objective unbalanced

binary SVM is constructed as:

$$\begin{aligned} \min_{\omega,\beta,\xi} \quad & \frac{1}{2}\|\omega\|^2 + C^+ \sum_{i\in I_+} \xi_i + C^- \sum_{i\in I_-} \xi_i, \\ \text{s.t.} \quad & y_i(\omega^T x_i + \beta) + \xi_i \geq 1, i \in I, \\ & \xi_i \geq 0, i \in I, \end{aligned} \tag{1.3}$$

where $\omega \in \mathbb{R}^l$, $\beta \in \mathbb{R}$, $\xi = (\xi_1, \xi_2, \cdots, \xi_n) \in \mathbb{R}^n$, $I_+ = \{i \mid i \in I \text{ and } y_i = 1\}$ and $I_- = \{i \mid i \in I \text{ and } y_i = -1\}$, C^+ and C^- are the different weights given to the classification errors (the penalties) of class "+" and class "-" objects, respectively.

1.2.2 Bi-objective binary SVM

The difficulty for the single-objective unbalanced binary SVM is to find proper values of C^+ and C^-. In many situations, the misclassification costs are imprecise or unknown. As we have mentioned for medical diagnosis problems, the cost of misclassifying a healthy person as ill is quite different from the one of misclassifying an ill patient as healthy. However, assigning concrete values to such costs is not simple. For this case, a more effective approach may be to use a bi-objective binary SVM method, as proposed in [Carrizosa and Martin-Barragan, 2006].

By maximizing the soft geometric margins $\widetilde{\rho}_1(\omega, \beta, \xi)$ and $\widetilde{\rho}_{-1}(\omega, \beta, \xi)$ [1], which have the penalties embedded in the objective functions, the bi-objective binary SVM is constructed as follows:

$$\begin{aligned} \max_{\omega,\beta,\xi} \quad & (\widetilde{\rho}_1(\omega, \beta, \xi), \widetilde{\rho}_{-1}(\omega, \beta, \xi)), \\ \text{s.t.} \quad & y_i(\omega^T x_i + \beta) + \xi_i \geq 1, i \in I, \\ & \xi_i \geq 0, i \in I, \end{aligned} \tag{1.4}$$

where $\omega \in \mathbb{R}^l$, $\beta \in \mathbb{R}$, $\xi = (\xi_1, \xi_2, \cdots, \xi_n) \in \mathbb{R}^n$,

$$\begin{aligned} \widetilde{\rho}_{x_i}(\omega, \beta, \xi) &= \frac{y_i(\omega^T x_i + \beta) + \xi_i}{\|(\omega, \xi)\|^*}, \\ \widetilde{\rho}_1(\omega, \beta) &= \min_{i\in I_+} \rho_{x_i}(\omega, \beta), \\ \widetilde{\rho}_{-1}(\omega, \beta) &= \min_{i\in I_-} \rho_{x_i}(\omega, \beta), \end{aligned}$$

and $\|\cdot\|^*$ stands for the weighted Euclidean norm, defined as:

$$\|(\omega, \xi)\|^* = \sqrt{\|\omega\|^2 + C^+ \sum_{i\in I_+} \xi_i^2 + C^- \sum_{i\in I_-} \xi_i^2}.$$

[1] Here, we have just reviewed the soft geometric margin version, but there exists also a hard geometric margin version, see [Carrizosa and Martin-Barragan, 2006]. In practice, it is more common to use the soft-margin SVMs, as it can be applied to the non-separable case and helps to avoid the overfitting problem. For the literature in the later property, we review only the soft-margin version to reduce the length of the presentation.

For the bi-objective programming problem problem (1.4), we need to get their Pareto-optimal solutions, instead of their optimal solutions. More generally, for multi-objective optimization problems, the goal is to identify their Pareto-optimal solutions. Following [Chinchuluun and Pardalos, 2007, Deb, 2001, Ehrgott, 2005], we can define the Pareto-optimal solutions and weakly Pareto-optimal solutions as follows: Given a general multi-objective problem,

$$\max_{\mu \in C}(f_1(\mu), f_2(\mu), \cdots, f_h(\mu)).$$

- A feasible solution μ^* is Pareto-optimal iff there does not exist another feasible solution $\mu \in C$ such that $f_i(\mu) \geq f_i(\mu^*)$ for all $i \in \{1, 2, \cdots, h\}$, and $f_j(\mu) > f_j(\mu^*)$ for at least one $j \in \{1, 2, \cdots, h\}$.
- A feasible solution μ^* is weakly Pareto-optimal iff there does not exist another feasible solution $\mu \in C$ such that $f_i(\mu) > f_i(\mu^*)$ for all $i \in \{1, 2, \cdots, h\}$.

In Theorem 1.1 the Pareto-optimal solutions of problem (1.4) are characterized.

Theorem 1.1. *[Carrizosa and Martin-Barragan, 2006] The set of Pareto-optimal solutions of the bi-objective problem (1.4) is given by*

$$W = \{(\lambda\omega_1, \lambda\beta, \lambda\xi_1) : |\beta - \beta_1| < 1, \lambda > 0\},$$

where $(\omega_1, \beta_1, \xi_1)$ *is the optimal solution of problem (1.5).*

$$\begin{aligned} \min_{\omega,\beta,\xi} \quad & \|\omega\|^2 + C^+ \sum_{i \in I_+} (\xi_i)^2 + C^- \sum_{i \in I_-} (\xi_i)^2, \\ s.t. \quad & y_i(\omega^T x_i + \beta) + \xi_i \geq 1, i \in I, \\ & \xi_i \geq 0, i \in I. \end{aligned} \tag{1.5}$$

Taking advantage of this theorem, it is possible to get different Pareto-optimal solutions of problem (1.4) from the solution of one quadratic problem problem (1.5). Thus, this bi-objective approach has a low computational cost, compared with other binary SVMs, while providing a large number of high quality solutions.

1.3 Single-objective Multi-class SVM

In real-life problems, we usually have more than two classes. Researchers have proposed several single-objective SVM models for multi-class classifications. These single-objective methods can be roughly grouped into two families. The first family constructs and combines several binary (two classes) classifiers, such as one-against-one, one-against-all and directed acyclic graph (DAG) SVMs, e.g. [Hsu and Lin, 2002, Kreßel, 1999, Platt et al., 2000, Vapnik, 1998]. Alternatively, all-together methods directly find a discriminant function by solving a single optimization problem, which attempts to classify all patterns into the corresponding classes, e.g.

[Bredensteiner and Bennett, 1999, Crammer and Singer, 2001, Hsu and Lin, 2002, Weston and Watkins, 1999].

For the results presented in this dissertation, we will assume that we have a training set $\{x_i\}_{i=1}^n \subseteq \mathbb{R}^l$, corresponding to $m(m \geq 3)$ different classes. Let $I = \{1, 2, \cdots, n\}$ denote the indexes of the training instances and $y_i \in G = \{1, ..., m\}$ denote the class membership of vector x_i.

1.3.1 All-together method

This approach aims to classify all the classes simultaneously, by solving one single optimization problem. As in binary SVMs, to find high classification-ability classifiers and overcome the over-fitting problem, it maximizes all the functional margins and minimizes all the misclassification errors at the same time. Each of these functional margins is defined as the distance between two support hyperplanes which are used to separate one class from the remaining ones. In the approach proposed by [Weston and Watkins, 1999], the classical binary SVM has been extended to multi-class classification problems as follows:

$$\begin{aligned} \min_{\omega,\beta,\xi} \quad & \frac{1}{2}\sum_{p\in G}[(\omega^p)^T\omega^p] + C\sum_{i\in I}\sum_{p\neq y_i}\xi_i^p \\ \text{s.t.} \quad & \omega_{y_i}^T x_i + \beta_{y_i} - [(\omega^p)^T x_i + \beta^p] + \xi_i^p \geq 2, p \in G \setminus y_i, i \in I, \\ & \xi_i^p \geq 0, p \in G \setminus y_i, i \in I, \end{aligned} \tag{1.6}$$

where $\omega^p \in \mathbb{R}^l$, $\omega = ({\omega^1}^T, {\omega^2}^T, \cdots, {\omega^m}^T)^T \in \mathbb{R}^{lm}$, $\beta = (\beta^1, \beta^2, \cdots, \beta^m)^T \in \mathbb{R}^m$ and $\xi \in \mathbb{R}^{(m-1)n}$ is a vector collecting all the $\xi_i^p, p \in G \setminus y_i, i \in I$.

The solution of this problem allows the definition of a decision function as:

$$f(x) = arg\max_{p\in G}[(\omega^p)^T x + \beta^p] \tag{1.7}$$

This single-objective approach is also known as an "all-together" method, in [Hsu and Lin, 2002]. This all-together method is limited to small problems due to the difficulty of solving large quadratic problems problem (1.6) when m is large.

1.3.2 One-against-all method

For multi-class classification, two commonly used single-objective SVM methods are the "one-against-all" and the "one-against-one" SVM methods. The one-against-all SVM method constructs m different classifiers. Each of these classifiers is defined from a classical binary SVM,

considering one of the m classes as class "+" and all the remaining classes as class "-". Specifically, the $p-th$ classifier is constructed as:

$$\begin{aligned} \min_{\omega^p,\beta^p,\xi_i^p,i\in I} \quad & \frac{1}{2}(\omega^p)^T(\omega^p) + C\sum_{i\in I}\xi_i^p, \\ \text{s.t.} \quad & (\omega^p)^T x_i + \beta^p + \xi_i^p \geq 1, \text{if } i \in I_p, \\ & -(\omega^p)^T x_i - \beta^p + \xi_i^p \geq 1, \text{if } i \notin I_p, \\ & \xi_i^p \geq 0, i \in I, \end{aligned} \tag{1.8}$$

where $\omega^p \in \mathbb{R}^l$, $\beta^p \in \mathbb{R}$, $\xi_i^p \in \mathbb{R}$ and $I_p = \{i \in I | y_i = p\}$.

After constructing the m classifiers, a decision function similar to (1.7) is used to decide the class membership.

1.3.3 One-against-one method

Alternatively, one-against-one methods construct $m(m-1)/2$ classifiers. Each of these classifiers is constructed from a classical binary SVM which considers only two of the m classes. The classifier which considers classes p and q, for example, is constructed as:

$$\begin{aligned} \min_{\omega^{pq},\beta^{pq},\xi^{pq}} \quad & \frac{1}{2}(\omega^{pq})^T(\omega^{pq}) + C\sum_{i\in I_p\cup I_q}\xi_i^{pq}, \\ \text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} + \xi_i^{pq} \geq 1, \text{if } i \in I_p, \\ & -(\omega^{pq})^T x_i - \beta^{pq} + \xi_i^{pq} \geq 1, \text{if } i \in I_q, \\ & \xi_i^{pq} \geq 0, i \in I_p \cup I_q, \end{aligned} \tag{1.9}$$

where $\omega^{pq} \in \mathbb{R}^l$, $\beta^{pq} \in \mathbb{R}$, and $\xi_i^{pq} \in \mathbb{R}$.

For this one-against-one SVM, a voting strategy is used as the decision rule after computing the classifiers. This voting strategy (also known as 'Max Wins') is defined as in [Hsu and Lin, 2002]: For instance x, if $\omega^{pq}x + \beta^{pq} > 0$, then the vote for the p-th class is increased by one. Otherwise, the vote for the q-th class is increased by one. After this procedure is completed, x is assigned to the class with the largest vote. In the case when two classes have identical numbers of votes, the one with smaller index is selected.

1.4 Multi-objective multi-class SVM

The aforementioned single-objective multi-class SVM methods have the same drawback as the classical binary SVM method: They do not consider differences in misclassification costs nor any a priori information available about the class distributions. Asymmetrically weighted single-objective SVMs provide a simple approach to overcome this drawback. Traditionally, a grid

search is used to find proper values for the tradeoff parameter in the binary SVMs. We can also use a grid-search method to find proper values of these weights for the multi-class classification problems. However, this approach requires solving the QPs associated to pairs of classes repeatedly, using different values of the parameters. As an alternative, we can use multi-objective approaches to address this problem. In [Tatsumi et al., 2011, 2007, 2009a,b, 2010, 2011, Tatsumi and Tanino, 2014], they extend some of the single-objective methods, such as all-together, one-against-all and one-against-one, to their multi-objective counterparts.

These multi-objective SVMs share many common features. They use the same classification rule: the value of $(\omega^p)^T x + \beta^p,\ p \in G$ is used to measure the degree of confidence that object x belongs to class p; then, x is assigned to the class with the highest degree of confidence. Also, their $m(m-1)$ objective functions and $(m-1)k$ constraints share the same structure. In general, the main properties sought in the definition of an SVM-based procedure are a high generalization ability and low training classification errors.

In the preceding multi-objective SVM proposals, pairwise geometric margins ρ^{pq} were used instead of functional margins $\sqrt{2}/\|\omega^p - \omega^q\|$, to achieve a higher generalization ability.

- The geometric margin between instances of class p and class q is defined as:

$$\rho^{pq} = \frac{\sigma^{pq}}{\|\omega^p - \omega^q\|}, q > p, p, q \in G,$$

  where

$$\sigma^{pq} = \min\left\{ \min_{i \in I_{pq}} \left\{(\omega^p - \omega^q)^T x_i + \beta^p - \beta^q\right\}, \min_{i \in I_{qp}} \left\{(\omega^q - \omega^p)^T x_i + \beta^q - \beta^p\right\} \right\}$$

  and $I_{pq} = \{i \in I_p | (\omega^p - \omega^q)^T x_i + \beta^p - \beta^q \geq 1\}, q > p, p, q \in G.$

When looking for high classification accuracies, these proposals minimize certain penalty functions, defined as weighted proportions of the sums of the auxiliary variables over the geometric margins, instead of minimizing each of the auxiliary variables.

- The penalty function for the misclassification errors for a pair of classes p and q is defined as:

$$\varsigma^{pq}(\sigma, \xi) = \frac{\eta^{pq}(\xi)/\|\omega^p - \omega^q\|}{\sigma^{pq}/\|\omega^p - \omega^q\|} = \frac{\eta^{pq(\xi)}}{\sigma^{pq}}, q > p, p, q \in G,$$

  where

$$\eta^{pq}(\xi) = \sum_{i \in I_p} \xi_i^{pq} + \sum_{i \in I_q} \xi_i^{qp}, q > p, p, q \in G.$$

In this way, it is not necessary to optimize a very large number of objective functions. For the constraints, all the objects are required to be correctly classified by all the associated discriminant hyperplanes, taking into account the auxiliary variables.

To solve these multi-objective problems it is usual to apply a $\varepsilon-$constraint method, which guarantees the weakly Pareto-optimality of the corresponding solutions by solving a large second-order cone-program (SOCP). This $\varepsilon-$constraint method will be reviewed in Section 1.4.4.

1.4.1 Multi-objective multi-class SVM based on an all-together method

In [Tatsumi et al., 2009b], the authors extend the all-together SVM method to its multi-objective counterpart. By maximizing all the pairwise geometric margins ρ^{pq} and minimizing the penalty functions ς^{pq}, they construct the multi-objective SVM based on an all-together method (MS2[2]) as follows:

$$\begin{aligned}
\max_{\omega,\beta,\sigma,\xi} \quad & \left(\frac{\sigma^{12}}{\|\omega^1-\omega^2\|}, \cdots, \frac{\sigma^{(m-1)m}}{\|\omega^{m-1}-\omega^m\|}, -\varsigma^{12}(\sigma,\xi), \cdots, -\varsigma^{(m-1)m}(\sigma,\xi) \right), \\
\text{s.t.} \quad & (\omega^p-\omega^q)^T x_i + (\beta^p-\beta^q) \geq \sigma^{pq} - \xi_i^{pq}, i \in I_p, q > p, p, q \in G, \\
& (\omega^q-\omega^p)^T x_i + (\beta^q-\beta^p) \geq \sigma^{pq} - \xi_i^{qp}, i \in I_q, q > p, p, q \in G, \\
& \sigma^{pq} \geq 1, q > p, p, q \in G, \\
& \xi_i^{pq} \geq 0, i \in I_p, q \neq p, p, q \in G,
\end{aligned} \tag{1.10}$$

where $\omega = ({\omega^1}^T, {\omega^2}^T, \cdots, {\omega^m}^T)^T \in \mathbb{R}^{ml}$, $\beta = (\beta^1, \beta^2, \cdots, \beta^m)^T \in \mathbb{R}^m$, $\sigma = (\sigma^{12}, \sigma^{13}, \cdots, \sigma^{(m-1)m})^T \in \mathbb{R}^{m(m-1)/2}$ and $\xi \in \mathbb{R}^{(m-1)n}$ collecting $\xi_i^{pq}, i \in I_p, q \neq p, p, q \in G$. Problem (1.10) has $m(l+1) + \frac{m-1}{2}(m+2n)$ decision variables and $\frac{m-1}{2}(4n+m)$ constraints with $m(m-1)$ objective functions.

1.4.2 Multi-objective multi-class SVM based on a one-against-all method

To alleviate the computational burden of MS2, Tatsumi et al. proposed a multi-objective multi-class SVM based on a one-against-all method (SM-OA) in [Tatsumi et al., 2010], which reduces the number of decision variables. They solve this SM-OA model in two phases: in the first phase, m classical binary SVMs have been solved as in a one-against-all method to compute a set of vectors $\bar{\omega}^p, p \in G$; in a second phase they define $\omega^p \equiv \alpha^p \bar{\omega}^p$, and the values for the scalars $\alpha^p, p \in G$ are obtained by solving:

$$\begin{aligned}
\max_{\alpha,\beta,\sigma,\xi} \quad & \left(\frac{\sigma^{12}}{\|\alpha^1\bar{\omega}^1-\alpha^2\bar{\omega}^2\|}, \cdots, \frac{\sigma^{(m-1)m}}{\|\alpha^{m-1}\bar{\omega}^{m-1}-\alpha^m\bar{\omega}^m\|}, -\varsigma^{12}, \cdots, -\varsigma^{(m-1)m} \right), \\
\text{s.t.} \quad & (\alpha^p\bar{\omega}^p-\alpha^q\bar{\omega}^q)^T x_i + (\beta^p-\beta^q) \geq \sigma^{pq} - \xi_i^{pq}, i \in I_p, q > p, p, q \in G \\
& (\alpha^q\bar{\omega}^q-\alpha^p\bar{\omega}^p)^T x_i + (\beta^q-\beta^p) \geq \sigma^{pq} - \xi_i^{qp}, i \in I_q, q > p, p, q \in G, \\
& \sigma^{pq} \geq 1, q > p, p, q \in G, \\
& \xi_i^{pq} \geq 0, i \in I_p, q \neq p, p, q \in G,
\end{aligned} \tag{1.11}$$

[2]We use MS2 to name this method following [Tatsumi et al., 2009b]. In all instances mentioned in this thesis we have chosen to keep the nomenclature used in the original reference.

where $\alpha = (\alpha^1, \alpha^2, \cdots, \alpha^m)^T \in \mathbb{R}^m$, $\beta = (\beta^1, \beta^2, \cdots, \beta^m)^T \in \mathbb{R}^m$, $\sigma = (\sigma^{12}, \sigma^{13}, \cdots, \sigma^{(m-1)m})^T \in \mathbb{R}^{m(m-1)/2}$ and $\xi \in \mathbb{R}^{(m-1)n}$ collecting $\xi_i^{pq}, i \in I_p, q \neq p, p, q \in G$.

1.4.3 Multi-objective multi-class SVM based on a one-against-one method

A hard-margin multi-objective multi-class SVM based on a one-against-one method (SM-OAO) was introduced in [Shoki Ishida, Keiji Tatsumi, 2012]. It is very easy to extend it to a soft-margin version as described for MS2 and SM-OA. As in previous cases, a soft-margin version is more useful than the corresponding hard-margin one in practice, because it can deal with the nonlinearly separable case and avoid the overfitting problem. The model problem (1.12) presents a soft-margin version for a multi-objective multi-class SVM based on a one-against-one method extend by us. This model again offers a reduction in the size of the decision variables of MS2. As in the case of SM-OA, they process SM-OAO in two phases. In the first phase, $m(m-1)/2$ classical binary SVMs are solved to obtain a set of vectors $\bar{\omega}^{pq}, q > p, p, q \in G$. Then, in a second phase they compute the optimal classifiers by solving:

$$\begin{aligned} \max_{\alpha,\beta,\sigma,\xi} \quad & \left(\frac{\sigma^{12}}{\|\omega^1 - \omega^2\|}, \cdots, \frac{\sigma^{(m-1)m}}{\|\omega^{m-1} - \omega^m\|}, -\varsigma^{12}, \cdots, -\varsigma^{(m-1)m} \right), \\ \text{s.t.} \quad & (\omega^p - \omega^q)^T x_i + (\beta^p - \beta^q) \geq \sigma^{pq} - \xi_i^{pq}, i \in I_p, q > p, p, q \in G \\ & (\omega^q - \omega^p)^T x_i + (\beta^q - \beta^p) \geq \sigma^{pq} - \xi_i^{qp}, i \in I_q, q > p, p, q \in G, \\ & \sigma^{pq} \geq 1, q > p, p, q \in G, \\ & \xi_i^{pq} \geq 0, i \in I_p, q \neq p, p, q \in G, \end{aligned} \tag{1.12}$$

where $\alpha = (\alpha^{12}, \alpha^{21}, \cdots, \alpha^{(m-1)m}, \alpha^{m(m-1)})^T \in \mathbb{R}^{m(m-1)}$, $\beta = (\beta^1, \beta^2, \cdots, \beta^m)^T \in \mathbb{R}^m$, $\sigma = (\sigma^{12}, \sigma^{13}, \cdots, \sigma^{(m-1)m})^T \in \mathbb{R}^{m(m-1)/2}$, $\xi \in \mathbb{R}^{(m-1)n}$ collecting $\xi_i^{pq}, i \in I_p, q \neq p, p, q \in G$ and

$$\omega^p = \sum_{q \neq p} \alpha^{pq} \bar{\omega}^{pq} \in \mathbb{R}^l, \text{ where } \bar{\omega}^{qp} = \bar{\omega}^{pq}, q > p, p, q \in G.$$

1.4.4 A ε-constraint method

These multi-objective approaches (SM2, MS-OA and MS-OAO) are frequently solved using a $\varepsilon-$constraint method. For example, a weakly Pareto-optimal solution of problem (1.10) can be

obtained by solving the following program:

$$\begin{aligned}
\max_{\omega,\beta,\sigma,\xi} \quad & \frac{\sigma^{rs}}{\|\omega^r - \omega^s\|}, \\
\text{s.t.} \quad & \frac{\sigma^{pq}}{\|\omega^p - \omega^q\|} \geq \varepsilon^{pq}, q > p, (p,q) \neq (r,s), p, q \in G, \\
& \varsigma^{pq} \leq \mu^{pq}, q > p, p, q \in G, \\
& (\omega^p - \omega^q)^T x_i + (\beta^p - \beta^q) \geq \sigma^{pq} - \xi_i^{pq}, i \in I_p, q > p, p, q \in G, \\
& (\omega^q - \omega^p)^T x_i + (\beta^q - \beta^p) \geq \sigma^{pq} - \xi_i^{qp}, i \in I_q, q > p, p, q \in G, \\
& \sigma^{pq} \geq 1, q > p, p, q \in G, \\
& \xi_i^{pq} \geq 0, i \in I_p, q \neq p, p, q \in G,
\end{aligned} \tag{1.13}$$

where (r,s), ε^{pq} and μ^{pq} are selected to ensure that the feasible region of problem (1.13) is not empty. A property of this solution is given in the following Theorems:

Theorem 1.2. *[Tatsumi et al., 2009b] Let $(\omega, \beta, \sigma, \xi)$ be an optimal solution of problem (1.13) for some (r,s). Then $(\omega, \beta, \sigma, \xi)$ is weakly efficient for problem (1.10).*

Theorem 1.3. *[Tatsumi et al., 2009b] $(\omega, \beta, \sigma, \xi)$ is Pareto optimal for problem (1.10) if and only if there exist values ε_{-rs} and μ such that $(\omega, \beta, \sigma, \xi)$ is optimal for problem (1.13) for any (r,s).*

However, problem (1.13) is still difficult to solve due to its fractional nonlinear objective function. In [Tatsumi et al., 2009b], they add a constraint $\sigma^{rs} = c^{rs}$ with an appropriate constant $c^{rs} \geq 1$, to obtain the following model:

$$\begin{aligned}
\max_{\omega,\beta,\sigma,\xi} \quad & \frac{c^{rs}}{\|\omega^r - \omega^s\|}, \\
\text{s.t.} \quad & \frac{\sigma^{pq}}{\|\omega^p - \omega^q\|} \geq \varepsilon^{pq}, q > p, (p,q) \neq (r,s), p, q \in G, \\
& \varsigma^{rs} = \frac{\eta^{rs}(\xi)}{c^{rs}} \leq \mu^{rs}, \\
& \varsigma^{pq} = \frac{\eta^{pq}(\xi)}{\sigma^{pq}} \leq \mu^{pq}, q > p, (p,q) \neq (r,s), p, q \in G, \\
& (\omega^r - \omega^s)^T x_i + (\beta^r - \beta^s) \geq c^{rs} - \xi_i^{rs}, i \in I_r, \\
& (\omega^s - \omega^r)^T x_i + (\beta^s - \beta^r) \geq c^{rs} - \xi_i^{sr}, i \in I_s, \\
& (\omega^p - \omega^q)^T x_i + (\beta^p - \beta^q) \geq \sigma^{pq} - \xi_i^{pq}, i \in I_p, q > p, (p,q) \neq (r,s), p, q \in G, \\
& (\omega^q - \omega^p)^T x_i + (\beta^q - \beta^p) \geq \sigma^{pq} - \xi_i^{qp}, i \in I_q, q > p, (p,q) \neq (r,s), p, q \in G, \\
& \sigma^{pq} \geq 1, q > p, (p,q) \neq (r,s), p, q \in G, \\
& \xi_i^{pq} \geq 0, i \in I_p, q \neq p, p, q \in G.
\end{aligned} \tag{1.14}$$

We have the following results, related the solutions of this problem:

Theorem 1.4. *[Tatsumi et al., 2009b] Let $(\hat{\omega}, \hat{b}, \hat{\sigma}, \hat{\xi})$ be an optimal solution of (problem (1.13)) and $c^{rs} = t\hat{\sigma}^{rs}$ for $t \geq 1$. If $(\omega^*, \beta^*, \sigma^{-rs*}, \xi^*)$ is an optimal solution of (problem (1.14)), then $(\omega^*, \beta^*, (\sigma^{-rs*}, c^{rs}), \xi^*)$ is optimal for (problem (1.13)).*

Theorem 1.5. *[Tatsumi et al., 2009b] If $(\omega^*, \beta^*, \sigma^*, \xi^*)$ is an optimal solution of (problem (1.13)), then for any $t \geq 1$, $(t\omega^*, t\beta^*, t\sigma^{-rs*}, t\xi^*)$ is an optimal solution of (problem (1.14)) with $c^{rs} = t\sigma^{rs*}$.*

From the description of these multi-objective approaches, it is possible to identify several challenges associated to their use. First of all, an efficient application of the $\varepsilon-$constraint method requires finding proper values of the constraint parameters ϵ^{-rs}, μ^{pq}, and objective parameters (r, s) and c^{rs}. This task is made even more complex when m is large. Second, after finding proper values of these parameters we have to solve large SOCPs as defined in problem (1.14), to obtain weakly Pareto-optimal solutions of the classification problem. Finally, as for multi-objective optimizations, we wish to find good approximations of the Pareto frontier. The solutions of problem (1.14) provide only weakly Pareto-optimal solutions. To get better approximations of the Pareto frontier, we have to repeatedly solve these large SOCPs, with very large associated computational costs.

1.5 Contributions

In this work, our focus is on SVM approaches for multi-class classifications. To generalize multi-class SVM models, we present two approaches taking into account the differences among misclassification costs. One is based on using a multi-objective SVM method, while the second one is formulated as a weighted single-objective SVM method.

Specifically, in Chapter 2 we introduce a multi-objective SVM (**PM**) for multi-class classification problems. This proposed multi-objective multi-class SVM is based on an adapted projection of the data (the reason why we call it a Projected Multi-objective SVM). Compared with the multi-objective SVMs described in the literature, **PM** provides higher quality classifiers using less computation. One interesting property of the proposed projection is that we are able to provide a theoretical guarantee for the Pareto-optimality solutions of **PM**, while the multi-objective SVMs in the literature apply a $\varepsilon-$constraint method, which can only guarantee the solutions' weakly Pareto-optimality. To approximate the Pareto-frontier, other multi-objective approaches face the big challenge of having to choose proper constraint parameters and solve large scale SOCPs repeatedly. As opposed to these properties, the Pareto-optimal solutions of **PM** are parallel to the solution of a single-objective QP, for any choice of parameters in the model. As a consequence, **PM** offers better quality solutions using significantly lower computational costs, compared with the multi-objective approaches in the literature. The properties of **PM**'s Pareto-optimal solutions also allow us to offer decision makers a large number of good choices, with quite reasonable computational costs, compared with other multi-class approaches such as the all-together method, one-against-all method, one-against-one method, MS2, SM-OA or

SM-OAO. Numerical experiments in Chapter 2 also give strong evidence for the advantage of **PM**.

In Chapter 3, we construct a weighted single-objective SVM to solve multi-class classification problems considering any differences among the misclassification costs. A difficulty of this approach is to find proper weights for the penalties in the objective functions. A grid search can be used to find these values, as most researchers have done for the binary classification problem. However, this grid search method needs to solve the weighted single-objective SVM repeatedly with different values of these parameters; this has a very high computational cost. Instead of using grid searches, it is possible to take advantage of the piecewise linearity of the optimal solutions of the single-objective SVM. In this thesis, we propose a partial parametric-path algorithm (**PPPA**) to overcome this difficulty. For **PPPA**, and given initial weights, we start the procedure by solving the corresponding single-objective SVM. Then, along a chosen search direction on the weights, a path of solutions for **PPPA** can be found by solving some linear equation systems, while finding the extremes of the segments in the piecewise-linear path by monitoring any changes in the active sets. We show that **PPPA** offers an efficient method to find proper values for the weight parameters. To improve the search space and to be able to explore in a systematic manner different weight vectors, we have combined **PPPA** and a variable neighborhood search method. Numerical experiments in Chapter 3 show the reliability and efficiency of **PPPA**.

In Chapter 4, concluding remarks of this thesis and some possible research lines are provided.

Chapter 2

A Projection Method for Multi-objective Multi-class SVM

2.1 Overview

As we have mentioned in Chapter 1 that the differences between the misclassification costs are necessary to be considered as they affect the classification performances of SVMs, [Akbani et al., 2004, Veropoulos et al., 1999]. A direct way to consider these differences is applying different weights to the misclassification penalties in a single-objective multi-class SVM for example an all-together SVM method. However, in many situations, the misclassification costs are fuzzy or unknown. So, in order to use a single-objective SVM approach to solve this problem, we need to find proper values for the weights of the misclassification penalties. Traditionally, grid searching is used to find proper values for these weights. This needs a lot of computation. We will focus on solving this difficulty in Chapter 3. Alternatively, we can use a multi-objective multi-class SVM approach to consider these differences.

In [Tatsumi et al., 2007], they used a multi-objective multi-class SVM for pattern recognition. Then based on one-against-all, one-against-one and all-together methods, they proposed a series of multi-objective SVMs to solve multi-class classification problems, e.g. [Tatsumi et al., 2011, 2009a,b, 2010, 2011, Tatsumi and Tanino, 2014]. As we have mentioned in Chapter 1, these multi-objective SVMs share many common features. On the other side, as we can see that MS2 has $m(l+1)+(m-1)(k+\frac{m}{2})$ variables. SM-OA and SM-OAO are multi-objective approaches introduced to reduce the number of the unknown variables. The number of variables in SM-OA is $\frac{1}{2}m(m+3)+(m-1)k$, while for SM-OAO we have $\frac{1}{2}m(3m-1)+(m-1)k$ variables. We can see that SM-OA has the smallest number of variables among these three multi-objective SVMs and if $l > m-1$, then SM-OAO has $m(l+1-m)$ fewer variables than MS2. Thus, both SM-OA and SM-OAO should be more computationally efficient than MS2, although as the optimal coefficients (ω^*, β^*) obtained from SM-OA (SM-OAO) are also feasible for MS2, their solutions will be no better than those provided by MS2.

However, it may be difficult and expensive to compute the complete set of Pareto-optimal solutions in many cases (e.g., large-scale optimization problems, complex structure of the Pareto-optimal solutions). In practice, the most common approach is to build an approximation to the Pareto-optimal solutions based on a limited number of solution values.

The aforementioned multi-objective multi-class SVMs have their drawbacks. Tatsumi et al. suggested to use the $\varepsilon-$constraint method, where the problem is transformed into a single-objective one, by selecting one of the objective functions while transforming all the rest into constraints, while setting a limit to their values. As these multi-objective SVMs have $m(m-1)$ objectives, we have to introduce $m(m-1)-1$ constraint parameters to define the constraints. When m is large, to find proper values for these constraint parameters may be a big challenge in practice (In their papers, they solved a large-scale QP (e.g. OS in [Tatsumi et al., 2010]) to get the proper values for these constraint parameters.). Each of the solutions approximating the Pareto-optimal set would be associated with a given set of values for these constraint parameters. The use of an $\varepsilon-$constraint method also introduces significant limitations to the solutions of these multi-objective SVMs. From [Ehrgott, 2005], we know that the $\varepsilon-$constraint method only guarantees weakly Pareto-optimal solutions. When m is large, to obtain a reasonable approximation to the Pareto-optimal solutions, beside the complication of finding proper values for these constraint parameters, we need to solve several computationally-expensive large-scale SOCPs (e.g. εSMOA2 in [Tatsumi et al., 2010]). It sums up that, with these aforementioned multi-objective multi-class SVMs using $\varepsilon-$constraint method, to get reasonable approximation to the Pareto-optimal solutions, we need a lot of computation to get the weakly Pareto-optimal solutions. Besides, they ignored that the cost of misclassifying class A objects as class B objects may be different from the cost of misclassifying class B objects as class A objects. For example, in medical diagnosis, it's known that the cost of misclassifying a healthy patient as ill is different from the one of misclassifying an ill patient as healthy. In medical diagnosis, as in many other applications, these differences need to be considered. For example, an investor may need a SVM which can separate high volatility shares from low volatility shares as accurately as possible, while it may be acceptable to misclassify some of the low volatility shares as high volatility shares.

In this chapter, we propose a practical multi-objective multi-class SVM that we denominate Projected Multi-objective SVM (**PM**), and which works in a higher dimensional space than the object space. Our aim is to address the main limitations that we have identified in the existing multi-objective SVM methods. Using **PM**, we are able to characterize the Pareto-optimal solutions of the problem by solving a single-objective QP problem once. Another advantage of our method is that, for large-scale problems, this single-objective QP problem can be decomposed into smaller subproblems in an efficient manner, significantly reducing its computational burden. Our proposal is also able to provide an approximation to the Pareto frontier with high out-of-sample quality, using limited computational cost. As a result, **PM** is both efficient and effective.

2.2 Projected multi-objective SVM

For simplicity we just consider the case of a linear classifier, since a nonlinear classifier can be considered as a linear classifier embedded in a richer object space. As in introduced in Chapter 1, for multi-class classification, the all-together method, one-against-all method and one-against-one method are the most commonly used single-objective methods, e.g. [Bredensteiner and Bennett, 1999, Crammer and Singer, 2001, Hsu and Lin, 2002, Kreßel, 1999, Vapnik, 1998, Weston and Watkins, 1999]. The single-objective all-together method needs to solve a large-scale optimization problem, so it is limited to small data sets [Hsu and Lin, 2002, Weston and Watkins, 1999]. The one-against-all method constructs m binary SVMs where each SVM classifies one of the classes versus the rest. Unbalances associated to the very different number of class objects in these binary SVMs may affect their classification accuracies and generalization abilities [Tatsumi et al., 2010, 2011]. In this regard, some experimental results seem to show that the one-against-all method may have a worse accuracy for some problems compared with the all-together and one-against-one methods [Hsu and Lin, 2002]. So probably, as suggested in [Hsu and Lin, 2002], the one-against-one method, which constructs $m(m-1)/2$ classifiers (discriminant hyperplanes), may be the most suitable single-objective approach for multi-class classification, compared with the all-together and one-against-all methods.

As the one-against-one method may be the most suitable single-objective approach for multi-class classification, we construct the discriminant hyperplanes as follows:

- The discriminant hyperplane to separate class p data against class q data is given by:
$$L^{pq}: \quad (\omega^{pq})^T x + \beta^{pq} = 0,\ q > p,\ p, q \in G,$$
where $\omega^{pq} \in \mathbb{R}^l$ and $\beta^{pq} \in \mathbb{R}\ q > p,\ p, q \in G$.

Ideally, we would like to have all class p objects lying above hyperplane L^{pq}, and all class q objects lying below L^{pq}. If there exist hyperplanes such that the training objects satisfy this ideal situation, we say that the training objects are linearly separable.

For the multi-class classification problems, when we have linearly separable training data, we require all the training instances been correctly classified and call the constructed SVMs as hard-margin ones. Considering about the over fitting problem and nonlinearly separable cases, we allow some misclassification errors in the training data. Minimizing all the classification errors is added to the objective functions and then we can construct the soft-margin multi-class SVMs.

In the following Section 2.2.1 and Section 2.2.2, we construct the hard-margin and soft-margin PMs with which we can find proper classifiers. After computing the classifiers from **PM**, we use majority voting (also known as 'Max Wins') to define our classification rule as in Section 1.3.3 .

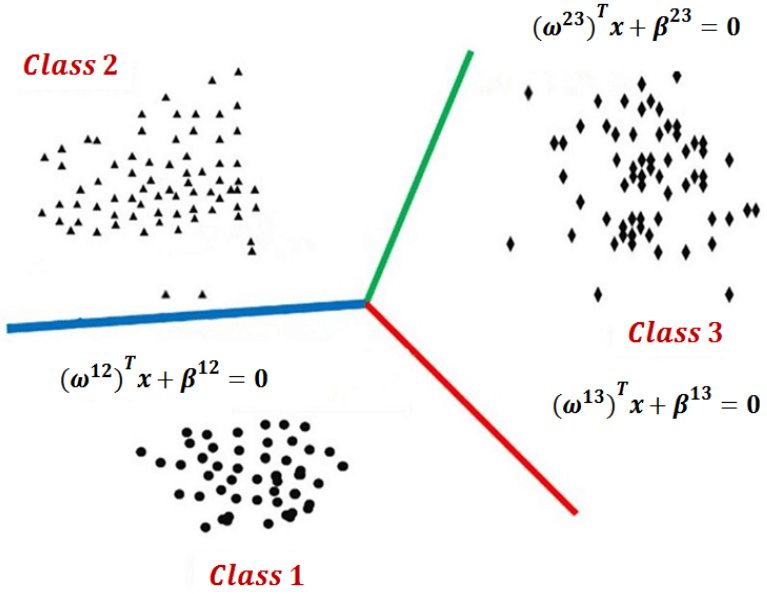


Figure 2.1: Linearly separable training objects from three classes

2.2.1 Hard-margin projected multi-objective multi-class SVM

Before we introduce our hard-margin **PM** method, we first construct a hard-margin single-objective multi-class SVM problem (P1) with which we can characterize the Pareto-optimal solution of the hard-margin **PM**.

$$\begin{aligned} \min_{\omega,\beta} \quad & \sum_{q>p,p,q\in G} \|\omega^{pq}\|^2, \\ \text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} \geq 1, \quad i \in I_p, q > p, p, q \in G, \\ & -(\omega^{pq})^T x_i - \beta^{pq} \geq 1, \quad i \in I_q, q > p, p, q \in G, \end{aligned} \tag{P1}$$

where $\omega = (\omega^{12^T}, \omega^{13^T}, \cdots, \omega^{(m-1)m^T})^T \in \mathbb{R}^{m(m-1)l/2}$ and $\beta = (\beta^{12}, \beta^{13}, \cdots, \beta^{(m-1)m})^T \in \mathbb{R}^{m(m-1)/2}$.

Note that problem (P1) is separable by pairs of classes. Hence, we have:

Property 2.1. *(ω_*, β_*) is optimal for problem (P1) if and only if the sub-vectors $(\omega_*^{pq}, \beta_*^{pq})$, $q > p$, $p, q \in G$ are optimal for the binary problems:*

$$\begin{aligned} \min_{\omega^{pq},\beta^{pq}} \quad & \frac{1}{2}\|\omega^{pq}\|^2, \\ s.t. \quad & (\omega^{pq})^T x_i + \beta^{pq} \geq 1, \ i \in I_p, \ q > p, \ p, q \in G, \\ & -(\omega^{pq})^T x_i - \beta^{pq} \geq 1, \ i \in I_q, \ q > p, \ p, q \in G, \end{aligned} \tag{2.1}$$

where $\omega^{pq} \in \mathbb{R}^l$ and $\beta^{pq} \in \mathbb{R}$.

In order to unify the denominators of the objectives so that for hard-margin **PM** method we can characterize its Pareto-optimal solutions, we introduce the following projection:

$$\Delta_x^{pq} = (\delta_x^{12^T}, \delta_x^{13^T}, \cdots, \delta_x^{(m-1)m^T})^T, \quad q > pp, q \in G,$$

with

$$\delta_x^{ij} = \begin{cases} x, \text{if } (i,j) = (p,q); \\ \mathbf{0}, \text{else.} \end{cases} \tag{2.2}$$

Then we can express hyperplane L^{pq} in the projected space as: $L^{pq} : \omega^T \Delta_x^{pq} + \beta^{pq} = 0$.

We define our hard geometric margin from object $x \in I_p$ to hyperplane L^{pq} as the Euclidean distance in the projected space:

$$\varrho_x^{pq}(\omega, \beta) = \frac{|(\omega)^T \Delta_x^{pq} + \beta^{pq}|}{\|\omega\|} = \frac{(\omega^{pq})^T x + \beta^{pq}}{\|\omega\|}, \quad x \in I_p, p \neq q, p, q \in G.$$

Notice that, in the separable case, we have all class p objects over hyperplane L^{pq}. So we have $(\omega^{pq})^T x + \beta^{pq} > 0$, for all $x \in I_p, p \neq q, p, q \in G$.

Now we define the hard geometric margin from class p to hyperplane L^{pq} as :

$$\varrho^{pq}(\omega, \beta) = \min_{x \in I_p} \varrho_x^{pq}(\omega, \beta), \quad p \neq q, p, q \in G.$$

In order to maximize all the pair-wise geometric margins $\varrho^{pq}(\omega, \beta)$, we can construct the hard-margin projected multi-objective SVM based on all-together method as:

$$\begin{aligned} \max_{\omega,\beta} \quad & \left(\varrho^{12}(\omega,\beta), \varrho^{21}(\omega,\beta), \cdots, \varrho^{(m-1)m}(\omega,\beta), \varrho^{m(m-1)}(\omega,\beta)\right) \\ \text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} > 0, \quad i \in I_p, q > p, p, q \in G, \\ & -(\omega^{pq})^T x_i - \beta^{pq} > 0, \quad i \in I_q, q > p, p, q \in G, \end{aligned} \tag{hard-margin \textbf{PM}}$$

where $\omega \in \mathbb{R}^{m(m-1)l/2}$ and $\beta \in \mathbb{R}^{m(m-1)/2}$.

For this multi-objective problemproblem (hard-margin **PM**), we define the following minimax weighted problem that provides Pareto-optimal solutions of problem (hard-margin **PM**):

$$\begin{aligned} \max_{\omega,\beta} \min \quad & \left(\varrho^{12}(\omega,\beta), \theta^{21}\varrho^{21}(\omega,\beta), \cdots, \theta^{(m-1)m}\varrho^{(m-1)m}(\omega,\beta), \theta^{m(m-1)}\varrho^{m(m-1)}(\omega,\beta)\right) \\ \text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} > 0, \quad i \in I_p, q > p, p, q \in G, \\ & -(\omega^{pq})^T x_i - \beta^{pq} > 0, \quad i \in I_q, q > p, p, q \in G. \end{aligned} \tag{2.3}$$

The above problem (2.3) will be a bridge for us to get the characterization of the Pareto-optimal solutions of problem (hard-margin **PM**). The following lemma establishes the relationship between problem (2.3) and problem (hard-margin **PM**). The values θ^{pq} can be seen as the proportions of the geometric margin ϱ^{12} over the geometric margins ϱ^{pq}.

Lemma 2.1. *(1) The optimal solution of problem (2.3) is weakly Pareto-optimal for problem (hard-margin **PM**);*
*(2) The weakly Pareto-optimal solutions of problem (hard-margin **PM**) are optimal for problem (2.3) given some specific values* $\boldsymbol{\theta} = (\theta^{21}, \cdots, \theta^{(m-1)m}, \theta^{m(m-1)}) > 0$.

The proof can be seen in Appendix A.1.

Problem (2.3) can be easily replaced with a quadratic problem. By solving that quadratic problem, we can characterize the weakly Pareto-optimal solutions of problem (hard-margin **PM**), as the following theorem shows.

Theorem 2.2. *The set of weakly Pareto-optimal solutions of problem (hard-margin **PM**) is :*

$$\{(\boldsymbol{\omega}, \boldsymbol{\beta}) = (\mu\omega_\theta, \mu\beta_\theta) \,|\mu > 0, \theta^{pq} > 0, p < q, p, q \in G\},$$

where $\theta^{12} = 1$, $\omega_\theta^{pq} = \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\omega_1^{pq}$ *and* $\beta_\theta^{pq} = \frac{\theta^{qp}-\theta^{pq}}{2\theta^{pq}\theta^{qp}} + \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\beta_1^{pq}$ *for all* $q > p, p, q \in G$, *with* (w_1, β_1) *being an optimal solution for problem (P1) .*

Proof. First, using the definition of the geometric margins, we can rewrite problem (2.3) as:

$$\begin{aligned}
\min_{\omega,\beta} \quad & \frac{\|\omega\|}{\min\left\{\min_{i\in I_1}(\omega^{12})^T x_i + \beta^{12}, \theta^{21}\min_{i\in I_2} -(\omega^{12})^T x_i - \beta^{12}, \cdots, \theta^{m(m-1)}\min_{i\in I_m} -(\omega^{(m-1)m})^T x_i - \beta^{(m-1)m}\right\}} \\
\text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} > 0, \quad i \in I_p, q > p, p, q \in G, \\
& -(\omega^{pq})^T x_i - \beta^{pq} > 0, \quad i \in I_q, q > p, p, q \in G.
\end{aligned} \tag{2.4}$$

We can see that (ω, β) is optimal for problem (2.4) iff $(\mu\omega, \mu b)$ is optimal for problem (2.4) for any $\mu > 0$. So we can standardize the denominator of the objective. Then we can solve the following problem to get the optimal solution of problem (2.4):

$$\begin{aligned}
\min_{\omega,\beta} \quad & \|\omega\| \\
\text{s.t.} \quad & \min\left\{\min_{i\in I_1}(\omega^{12})^T x_i + \beta^{12}, \theta^{21}\min_{i\in I_2}(-\omega^{12})^T x_i - \beta^{12}, \cdots, \theta^{m(m-1)}\min_{i\in I_m}(-\omega^{(m-1)m})^T x_i - \beta^{(m-1)m}\right\} = 1.
\end{aligned} \tag{2.5}$$

Easily we can see the above problem (2.5) is equivalent to

$$\begin{aligned}
\min_{\omega,\beta} \quad & \|\omega\| \\
\text{s.t.} \quad & \theta^{pq}[(\omega^{pq})^T x_i + \beta^{pq}] \geq 1, \quad i \in I_p, q > p, p, q \in G, \\
& \theta^{qp}[-(\omega^{pq})^T x_i - \beta^{pq}] \geq 1, \quad i \in I_q, q > p, p, q \in G.
\end{aligned} \tag{2.6}$$

Problem (2.6) is equivalent to:

$$\begin{aligned} \min_{\omega,\beta} \quad & \|\omega\|^2 \\ \text{s.t.} \quad & \theta^{pq}[(\omega^{pq})^T x_i + \beta^{pq}] \geq 1, \quad i \in I_p, q > p, p, q \in G, \\ & \theta^{qp}[-(\omega^{pq})^T x_i - \beta^{pq}] \geq 1, \quad i \in I_q, q > p, p, q \in G. \end{aligned} \tag{2.7}$$

As the objective function of problem (2.7) is strictly convex, we can see that the optimal solution ω_θ is unique. Besides, considering that the objective of problem (2.7) is quadratic (positive definite) and the constraints are affine functions, KKT conditions are necessary and sufficient for optimality. The KKT conditions for problem (2.7) are:

$$\begin{aligned} & 2\omega_\theta^{pq} = \theta^{pq} \sum_{i \in I_p} \lambda_{\theta i}^{pq} x_i - \theta^{qp} \sum_{i \in I_q} \lambda_{\theta i}^{qp} x_i, \quad q > p, p, q \in G, \\ & \theta^{pq} \sum_{i \in I_p} \lambda_{\theta i}^{pq} - \theta^{qp} \sum_{i \in I_q} \lambda_{\theta i}^{qp} = 0, \quad q > p, p, q \in G, \\ & \lambda_{\theta i}^{pq} [\theta^{pq} (\omega_\theta^{pq})^T x_i + \theta^{pq} \beta_\theta^{pq} - 1] = 0, \quad i \in I_p, q > q, p, q \in G, \\ & \lambda_{\theta i}^{qp} [\theta^{qp} (-\omega_\theta^{pq})^T x_i - \theta^{qp} \beta_\theta^{pq} - 1] = 0, \quad i \in I_q, \ q > p, \ p, q \in G, \\ & \lambda_{\theta i}^{pq} \geq 0, \quad i \in I_p, p \neq q, p, q \in G, \\ & \theta^{pq} [(\omega_\theta^{pq})^T x_i + \beta_\theta^{pq}] \geq 1, \quad i \in I_p, q > p, p, q \in G, \\ & \theta^{qp} [-(\omega_\theta^{pq})^T x_i - \beta_\theta^{pq}] \geq 1, \quad i \in I_q, q > p, p, q \in G \end{aligned} \tag{2.8}$$

From these KKT conditions, we can see that $(\lambda_\theta^{pq}, \lambda_\theta^{qp}) \neq 0, q > p, p, q \in G$. Without loss of generality, we can say that, for each $p, q \in G$ with $q > p$, there exist some $x_\theta^{pq} \in I_p$ such that $\lambda_{\theta x}^{pq} \neq 0$. Then we get

$$\beta_\theta^{pq} = \frac{1}{\theta^{pq}} - (\omega_\theta^{pq})^T x_\theta^{pq}, \quad q > p, p, q \in G.$$

So we can see that the set of optimal solutions for problem (2.7) is nonempty. Considering the convexity of the objective function, we have that problem (2.7) has a unique optimal solution.

(ω_1, β_1) is optimal for problem (P1). Let λ_1 be the corresponding KKT multiplier vector. Then take:

$$\begin{aligned} \omega_\theta^{pq} &= \frac{\theta^{pq} + \theta^{qp}}{2\theta^{pq}\theta^{qp}} \omega_1^{pq}, \quad q > p, \ p, q \in G, \\ \beta_\theta^{pq} &= \frac{\theta^{qp} - \theta^{pq}}{2\theta^{pq}\theta^{qp}} + \frac{\theta^{qp} + \theta^{pq}}{2\theta^{pq}\theta^{qp}} \times \beta_1^{pq}, \quad q > p, q, p \in G, \\ \lambda_{\theta i}^{pq} &= \frac{\theta^{pq} + \theta^{qp}}{2\theta^{pq}\theta^{qp}} \times \frac{1}{\theta^{pq}} \lambda_{1i}^{pq}, \quad i \in I_p, \ p \neq q, p, q \in G. \end{aligned} \tag{2.9}$$

Then $(\omega_\theta, \beta_\theta)$ will be the unique optimal solution of problem (2.7), since it satisfies the KKT conditions. Then, for any $\mu > 0$ we have that $(\mu\omega_\theta, \mu\beta_\theta)$ is optimal for problem (2.4). Using Lemma 2.1, we conclude that $(\mu\omega_\theta, \mu\beta_\theta)$ is weakly Pareto-optimal for problem (hard-margin **PM**). □

After characterizing these weakly Pareto-optimal solutions of problem (hard-margin **PM**), we try to identify its Pareto-optimal solutions. We now show that these weakly Pareto-optimal solutions will also be Pareto-optimal for problem (hard-margin **PM**).

Corollary 2.3. *The Pareto-optimal solution set of problem (hard-margin **PM**) will be:*

$$\{(\boldsymbol{\omega}, \boldsymbol{\beta}) = (\mu\omega_\theta, \mu\beta_\theta) \,|\mu > 0, \theta^{pq} > 0, p < q, p, q \in G\},$$

where $\theta^{12} = 1$, $\omega_\theta^{pq} = \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\omega_1^{pq}$ *and* $\beta_\theta^{pq} = \frac{\theta^{qp}-\theta^{pq}}{2\theta^{pq}\theta^{qp}} + \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\beta_1^{pq}$ *for all* $q > p, p, q \in G$, *with* (w_1, β_1) *being optimal to problem* (*P*1) .

Proof. From the definitions of Pareto-optimal and weakly Pareto-optimal solutions, we know that the Pareto-optimal solutions will also be weakly Pareto-optimal. So we only need to prove that the weakly Pareto-optimal solutions of problem (hard-margin **PM**) will also be Pareto-optimal.

Let (ω_*, β_*) be a weakly Pareto-optimal solution of problem (hard-margin **PM**). Then, there exist some $\theta > 0$ and $\mu > 0$ such that $(\mu\omega_*, \mu\beta_*)$ will be optimal for problem (2.7). Suppose (ω_*, β_*) is not Pareto-optimal for problem (hard-margin **PM**). For any $\mu > 0$ we have $\varrho^{pq}(\omega, \beta) = \varrho^{pq}(\mu\omega, \mu b)$. So $(\mu\omega_*, \mu\beta_*), \forall\mu > 0$ will not be Pareto-optimal for problem (hard-margin **PM**). Then there exist (ω_0, β_0) such that:

$$\varrho^{pq}(\omega_0, \beta_0) \geq \varrho^{pq}(\mu\omega_*, \mu\beta_*), \quad p \neq q, p, q \in G, \tag{2.10}$$

and at least one $(i, j), i \neq j, i, j \in G$, such that $\varrho^{ij}(\omega_0, \beta_0) > \varrho^{ij}(\mu\omega_*, \mu\beta_*)$.

Without loss of generality, we can take $\|\omega_0\| = \|\mu\omega_*\|$. Then we have:

$$(\omega_0^{pq})^T x + \beta_0^{pq} \geq (\mu\omega_*^{pq})^T x + \mu\beta_*^{pq}, \quad x \in I_p, p \neq q, p, q \in G.$$

As $(\mu\omega_*, \mu\beta_*)$ is optimal for problem (2.7), we have that (ω_0, β_0) is also feasible for problem (2.7). As $\|\omega_0\| = \|\mu\omega_*\|$, we can say that (ω_0, β_0) is optimal for problem (2.7). Since problem (2.7) has a unique optimal solution, we must have $\omega_0 = \mu\omega_*$, $\beta_0 = \mu\beta_*$. Thus, we have:

$$\varrho^{pq}(\omega_0, \beta_0) = \varrho^{pq}(\mu\omega_*, \mu\beta_*), \quad \forall p \neq q, p, q \in G.$$

This contradicts our assumption that problem (2.10) has at least one strict inequality. We then conclude that (ω_*, β_*) is Pareto-optimal for problem (hard-margin **PM**). □

2.2.2 Soft-margin projected multi-objective all-together

In Section 2.2.1 we have introduced the hard-margin **PM** problem. As before, in order to consider the overfitting problem and nonlinearly separable training objects, we derive a soft-margin

PM. We first introduce the following soft-margin sing-objective multi-class SVM problem (P2) with which we can characterize the Pareto-optimal solutions of the soft-margin **PM**.

$$\begin{aligned} \min_{\omega,\beta,\xi} \quad & \|\omega\|^2 + \sum_{q\neq p,p,q\in G} c^{pq} \sum_{i\in I_p} (\xi_i^{pq})^2, \\ \text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} + \xi_i^{pq} \geq 1, \quad i \in I_p, q > p, p, q \in G, \\ & -(\omega^{pq})^T x_i - \beta^{pq} + \xi_i^{qp} \geq 1, \quad i \in I_q, q > p, p, q \in G, \\ & \xi_i^{pq} \geq 0, i \in I_p, q \neq p, p, q \in G, \end{aligned} \tag{P2}$$

where $\omega = ({\omega^{12}}^T, {\omega^{13}}^T, \cdots, {\omega^{(m-1)m}}^T)^T \in \mathbb{R}^{m(m-1)l/2}$, $\beta = (\beta^{12}, \beta^{13}, \cdots, \beta^{(m-1)m})^T \in \mathbb{R}^{m(m-1)/2}$ and $\xi \in \mathbb{R}^{(m-1)n}$ collecting all the $\xi_i^{pq}, i \in I_p, q \neq p, p, q \in G$.

The above program (P2) is a strictly convex quadratic problem, so it has an unique optimizer.

And we have:

Property 2.2. *$(\omega_*, \beta_*, \xi_*)$ is optimal for problem (P2) if and only if the sub-vectors $(\omega_*^{pq}, \beta_*^{pq}, \xi_*^{pq}, \xi_*^{qp})$, $q > p$, $p, q \in G$ are optimal for the binary problems:*

$$\begin{aligned} \min_{\omega^{pq},\beta^{pq},\xi^{pq},\xi^{qp}} \quad & \frac{1}{2}\|\omega^{pq}\|^2 + c^{pq} \sum_{i\in I_p} (\xi_i^{pq})^2 + c^{qp} \sum_{i\in I_q} (\xi_i^{qp})^2, \\ s.t. \quad & (\omega^{pq})^T x_i + \beta^{pq} + \xi_i^{pq} \geq 1, \ i \in I_p, \\ & -(\omega^{pq})^T x_i - \beta^{pq} + \xi_i^{qp} \geq 1, \ i \in I_q, \\ & \xi_i^{pq} \geq 0, i \in I_p, \xi_i^{qp} \geq 0, i \in I_q, \end{aligned} \tag{2.11}$$

where $\omega^{pq} \in \mathbb{R}^l$, $\beta^{pq} \in \mathbb{R}$ and $\xi_i^{pq}, \xi_i^{qp} \in \mathbb{R}$.

We also need to properly define the geometric margins so that we can characterize the Pareto-optimal solutions for the resulting soft-margin **PM**. We are interested in a SVM formulation whose objective functions integrate both the maximization of the geometric margins and the minimization of the misclassification errors. We incorporate the misclassification errors into our model by redefining the geometric margins after embedding the slack variables (as measures of misclassification) into them. We consider the following projection:

$$\Delta_{\xi x}^{pq} = ({\delta_{\xi x}^{12}}^T, {\delta_{\xi x}^{21}}^T, \cdots, {\delta_{\xi x}^{m(m-1)}}^T)^T, \quad q > p, p, q \in G,$$

where

$$\delta_{\xi x}^{rs} = \begin{cases} \frac{1}{\sqrt{c^{pq}}}\mathbf{e}_i & \text{if } (r,s) = (p,q) \text{ and } x \text{ is the } i\text{-th object in class } p, \\ \mathbf{0} & \text{if } (r,s) \neq (p,q), r \neq s, r, s \in G, \end{cases}$$

and $\mathbf{e}_i$ is the i-th unit vector.

In the projected space we can construct the hyperplane classifying class p objects against class q objects as:

$$L^{pq} : (\omega, \sqrt{C}\xi)^T (\Delta_x^{pq}, \Delta_{\xi x}^{pq}) + \beta^{pq} = 0, \quad q > p, p, q \in G,$$

where, $\sqrt{C}\xi = (\sqrt{c^{12}}\xi^{12}, \sqrt{c^{21}}\xi^{21}, \cdots, \sqrt{c^{m(m-1)}}\xi^{m(m-1)})$ and Δ_x^{pq} defined as in Section 2.2.1.

We define the soft geometric margin from object x to hyperplane L^{pq} as the Euclidean distance in the projected space.

- The soft geometric margin from object $x \in I_p$ to hyperplane L^{pq} is:

$$\bar{\varrho}_x^{pq}(\omega, \sqrt{C}\xi, \beta) = \frac{|(\omega, \sqrt{C}\xi)^T(\Delta_x^{pq}, \Delta_{\xi x}^{pq}) + \beta^{pq}|}{\|(\omega, \sqrt{C}\xi)\|} = \frac{(\omega^{pq})^T x + \xi^{pq} + \beta^{pq}}{\|(\omega, \sqrt{C}\xi)\|}, \quad x \in I_p, p \neq q, p, q \in G.$$

- The soft geometric margin for class p objects against class q objects is:

$$\bar{\varrho}^{pq}(\omega, \sqrt{C}\xi, \beta) = \min_{x \in I_p} \bar{\varrho}_x^{pq}(\omega, \sqrt{C}\xi, \beta), \quad p \neq q, p, q \in G.$$

By maximizing all the soft geometric margins defined with the slack variables embedded, we can formulate the soft-margin **PM** as follows:

$$\begin{aligned} \max_{\omega,\beta,\xi} \quad & \left(\bar{\varrho}^{12}(\omega,\beta), \bar{\varrho}^{21}(\omega,\beta), \cdots, \bar{\varrho}^{(m-1)m}(\omega,\beta), \bar{\varrho}^{m(m-1)}(\omega,\beta)\right) \\ \text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} + \xi_i^{pq} > 0, \quad i \in I_p, q > p, p, q \in G, \\ & -(\omega^{pq})^T x_i - \beta^{pq} + \xi_i^{qp} > 0, \quad i \in I_q, q > p, p, q \in G, \\ & \xi_i^{pq} \geq 0, \quad i \in I_p, p \neq q, p, q \in G, \end{aligned} \qquad \text{(soft-margin PM)}$$

where $\omega \in \mathbb{R}^{m(m-1)l/2}$, $\beta \in \mathbb{R}^{m(m-1)/2}$ and $\xi \in \mathbb{R}^{(m-1)n}$.

By applying procedures similar to the ones used in Theorem 2.2 and Corollary 2.3, we can characterize the weakly Pareto-optimal and Pareto-optimal solutions for (soft-margin **PM**).

Theorem 2.4. *The set of weakly Pareto-optimal solutions for (soft-margin **PM**) is :*

$$\{(\boldsymbol{\omega}, \boldsymbol{\beta}, \boldsymbol{\xi}) = (\mu\omega_\theta, \mu\beta_\theta, \mu\xi_\theta) \,|\mu > 0, \theta > 0\},$$

where $\theta^{12} = 1$, $\omega_\theta^{pq} = \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\omega_1^{pq}$, $\beta_\theta^{pq} = \frac{\theta^{qp}-\theta^{pq}}{2\theta^{pq}\theta^{qp}} + \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\beta_1^{pq}$ *for all* $q > p$ *and* $\xi_\theta^{pq} = \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\xi_1^{pq}$ *for* $q \neq p, p, q \in G$, *with* $(\omega_1, \beta_1, \xi_1)$ *being optimal to (P2).*

The proof is similar to the proof of Theorem 2.2. The details can be found in Appendix A.2.

Corollary 2.5. *The Pareto-optimal solution set of (soft-margin **PM**) will be:*

$$\{(\boldsymbol{\omega}, \boldsymbol{\beta}, \boldsymbol{\xi}) = (\mu\omega_\theta, \mu\beta_\theta, \mu\xi_\theta) \,|\mu > 0, \theta > 0\},$$

where $\theta^{12} = 1$, $\omega_\theta^{pq} = \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\omega_1^{pq}$, $\beta_\theta^{pq} = \frac{\theta^{qp}-\theta^{pq}}{2\theta^{pq}\theta^{qp}} + \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\beta_1^{pq}$ *for all* $q > p$ *and* $\xi_\theta^{pq} = \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\xi_1^{pq}$ *for* $q \neq p, p, q \in G$, *with* $(\omega_1, \beta_1, \xi_1)$ *being optimal to* (P2).

The proof of this result is identical to the proof for Corollary 2.3.

From Corollary 2.5(Corollary 2.3), we can see that the Pareto-optimal solutions of soft-margin **PM**(hard-margin **PM**) are based on the solutions of the quadratic optimization problem P2(P1). This problem has the added advantage of being decomposable into binary classification problems, with the corresponding computational advantages when the problem size increases.

2.3 Computational experiments

We have conducted several computational experiments to test the practical behavior of the proposed procedure **PM**[1], when compared to several alternative multi-class SVMs described in the literature. These experiments have been conducted on the following datasets: IRIS, WINE, SEEDS, VEHICLE, CAR (Car Evaluation), GLASS, SCC (Synthetic Control Chart Time Series) and CTG (Cardiotocography, raw data). All of them are available in the UCI Machine Learning Repository. A summary of the information for these data sets is listed in Table 2.1.

Table 2.1: Data set description

Data set	size of the data set	No. of Dim.	No. of classes
IRIS	150	4	3
WINE	178	13	3
SEEDS	210	7	3
VEHICLE	846	18	4
CAR	1728	6	4
GLASS	214	9	6
SCC	600	60	6
CTG	2126	35	10

The first group of experiments compares **PM** with other multi-class SVMs (both the single-objective and multi-objective methods) in terms of their training classification accuracies, testing classification accuracies and training time. We consider that one method is superior to another when it has higher accuracies and lower computational costs. In this chapter, we compare the performances of the all-together method (AT) [Vapnik, 1998], one-against-all method (OAA) [Hsu and Lin, 2002], one-against-one method (OAO) [Kreßel, 1999], MS2, SM-OA, SM-OAO and **PM**.

Table 2.2 shows these measures averaged over 100 replications of random splittings of the dataset into a training sample (80%) used to compute the classifiers and a testing sample (20%) used to compute their testing accuracy.

We have chosen the parameters required by the different methods in the following way: for AT, OAA and OAO, we set the trade-off parameters to $C = 1$; for MS2, SM-OA and SM-OAO, we take $c^{rs} = 10$, $(r, s) = (1, 2)$ and fix (ε^{-rs}, μ) as the authors suggested in [Tatsumi et al.,

[1]In the experiments, we just use the soft-margin **PM**, because it is available for linearly separable and non linearly separable cases.

2009a, 2010, 2011, Tatsumi and Tanino, 2014]; for **PM**, we take $c^{pq} = 1,\ q \neq p,\ p, q \in G$. For every replication of each dataset, we solve all the SVM methods (AT, OAA, OAO, MS2, SM-OA and SM-OAO) once and record the corresponding training classification accuracies, testing classification accuracies and training time. For **PM**, we solve (P2) once and choose the best performance (accuracy) from 100 Pareto-optimal solutions of **PM** obtained randomly using Corollary 2.5.

Table 2.2: Mean results to compare the performances of the multi-class SVMs

		AT	OAA	OAO	MS2	SM-OA	SM-OAO	**PM**
IRIS	tr.ac	0.9859	0.9513	0.9871	0.6667	0.9845	0.9838	**0.9902**
	te.ac	0.9773	0.9387	0.9753	0.6667	0.9750	0.9723	**0.9870**
	tr.t(s)	**1.0001**	3.0293	3.0435	2.0449	4.0132	4.0770	*1.2147*
WINE	tr.ac	0.9865	0.9965	0.9959	0.8910	0.9978	0.5122	**0.9995**
	te.ac	0.9415	0.9594	0.9500	0.8718	0.9568	0.4982	**0.9741**
	tr.t(s)	**1.7463**	3.9034	3.8853	3.0276	5.1931	5.3379	*1.8346*
SEEDS	tr.ac	0.9518	0.9416	0.9360	0.9412	0.9421	0.8978	**0.9570**
	te.ac	0.9310	0.9255	0.9150	0.9112	0.9212	0.8886	**0.9493**
	tr.t(s)	**0.6690**	1.9245	1.8638	1.6396	2.4636	2.6796	*0.6835*
VEHICLE	tr.ac	0.8404	0.8208	0.8481	0.8392	0.8106	0.6585	**0.8525**
	te.ac	0.7984	0.7896	0.7859	0.8014	0.7789	0.6436	**0.8016**
	tr.t(s)	**2.9827**	9.3945	10.2526	24.7038	12.2850	12.4967	*3.4189*
CAR	tr.ac	0.8910	0.8548	0.9047	0.8829	0.8744	0.8711	**0.9072**
	te.ac	0.8845	0.8463	0.8959	0.8788	0.8669	0.8681	**0.9059**
	tr.t(s)	**0.6279**	1.4334	1.9986	37.3476	2.5928	2.9248	*1.1151*
GLASS	tr.ac	0.6807	0.6478	0.6866	0.6081	0.6366	0.3556	**0.7007**
	te.ac	0.6387	0.5810	0.6302	0.5665	0.5972	0.3375	**0.6617**
	tr.t(s)	**0.8233**	4.0339	9.7277	2.6385	4.8015	10.4632	*1.0687*
SCC	tr.ac	1	0.9900	1	0.51	1	0.9907	**1**
	te.ac	0.9827	0.9353	0.9865	0.5037	0.9426	0.9549	**0.9926**
	tr.t(s)	15.0726	5.0784	17.1662	25.7054	6.1961	13.8103	**2.2722**
CTG	tr.ac	1	1	1	1	1	0.6770	**1**
	te.ac	0.9999	**1**	0.9743	0.9997	0.9999	0.6717	*0.9821*
	tr.t(s)	**14.0502**	15.9397	55.0768	6410.388	41.3981	91.1074	18.1388

Note: The results marked in bold are the best average results. The results marked in italics are the second best average results, when they have been obtained by **PM**. In the following tables, we use the same way to highlight the results.

[1] tr.ac= training classification accuracy,

[2] te.ac= testing classification accuracy,

[3] tr.t(s)= training time measured in seconds.

From Table 2.2, we can see that **PM** performs best for most data sets. In particular, we can see that **PM** always shows the best mean training classification accuracies. And except for the CTG dataset, **PM** also gives the best mean testing classification accuracies. Even for the CTG dataset, **PM** achieves the second-best mean testing classification accuracy. Considering the mean training time, we can see that **PM** requires only slightly more time than the best value, that of single-objective method AT. For the SCC dataset, **PM** takes the shortest time.

Nevertheless, as our main goal is to find a reasonably detailed representation of the set of all Pareto-optimal solutions of the classification problems, these measures are not the ones most suitable for comparing the performances of the different multi-objective SVMs. We have already mentioned that for MS2, SM-OA and SM-OAO, we can obtain the weakly Pareto-optimal solutions by using the $\varepsilon-$constraint method, while for **PM**, we are able to provide a characterization for its Pareto-optimal solutions corresponding to a specific set of parameters. In both cases it is too expensive to determine all the Pareto-optimal solutions, as the structure of this set is very complex, particularly for high dimensions. Our aim is to find a good and efficient approximation to the Pareto-optimal solution sets. In this second set of experiments, we compare the different multi-objective methods with respect to the quality of their approximations for the Pareto-optimal solution sets: we say that a method outperforms another when it approximates the Pareto-optimal solution set better than the other. In this chapter, we use the epsilon and hypervolume indicators, defined in terms of the test accuracies as objectives to measure the performance of these multi-objective SVMs. These indicators have the important property of being Pareto compliant (whenever an approximation set A is preferable to B with respect to weak Pareto dominance, the indicator value for A should be at least as good as the indicator value for B [Fonseca and Knowles, 2005, Zitzler et al., 2003]. Following [Fonseca and Knowles, 2005],

- The hypervolume indicator $I_H(A)$ calculates the proportion of the objective space that is weakly dominated by an approximation set of Pareto-optimal solutions A.
- The epsilon indicator is defined as $I_{\epsilon+} = \inf_{\epsilon\in\mathbb{R}}\{\forall z^2 \in R, \exists z^1 \in A \text{ such that } z^1 \preceq_{\epsilon+} z^2\}$, where R is an reference set.

For the hypervolume indicator, we take the objective space as the hypercube which contains all possible testing classification accuracies $(a_1, a_2, \ldots, a_m)$. a_i is the testing classification accuracy of class i. Based on the definition of the hypervolume indicator, it holds that the values of the hypervolume indicators will be in $[0, 1]$, and the method which has the largest hypervolume indicator (closest to 1) outperforms the others.

For the epsilon indicator, we select the reference set R as $\{(1, \ldots, 1)\} \subset \mathbb{R}^m$, which corresponds to the ideal test accuracy. From the definition of the epsilon indicator, it holds that the values of the epsilon indicators will be in $[-1, 0]$ and the method which has the largest value of its epsilon indicator (closest to 0) outperforms the others.

To obtain a more stable performance measure, we have applied the procedure described below; in each replication we have selected a different subset of 80% of our observations as a training sample and the remaining observations as our testing sample. We have used Matlab R2014a and Mosek 7 to solve the optimization problems. As Mosek can give us 'unknown' solutions, we have only kept the 'optimal' and 'near-optimal' solutions and discarded the other results, to ensure the reliability of the results.

Step 1: For $i = 1, \cdots, 50$, we repeat:

- Step 1.1: Arrange the objects in a random order. Choose the last 20% objects as the testing objects and leave the rest as the training objects.
- Step 1.2: Do:
  - 1.2.a: If the method used is MS2, SM-OA or SM-OAO, then
    - Step 1.2.a1: As in [Tatsumi et al., 2009a, 2010, 2011, Tatsumi and Tanino, 2014], we obtain the parameters $(\varepsilon_0^{-rs}, \mu_0)$ by solving corresponding single-objective SVMs such as OS in [Tatsumi et al., 2010].
    - Step 1.2.a2: We take $(r, s) = (1, 2)$ and $c = 10$. Then we uniformly choose 100 different values for (ε^{-rs}, μ) with ε^{-rs} from $[0.1\varepsilon_0^{-rs}, 2\varepsilon_0^{-rs}]$ and μ from $[\mu_0 + 0.01, \mu_0 + 50]$. Then, we solve 100 SOCPs such as εSMOA2 [Tatsumi et al., 2010] defined from each value of (ε^{-rs}, μ). As we only keep the 'optimal' and 'near-optimal' solutions, we get at most 100 weakly Pareto-optimal solutions for the corresponding multi-objective method.
  - 1.2.b: If the method used is **PM**, then
    - Step 1.2.b1: We use a 10-fold cross validation method to choose the values $c^{pq},\ q \neq p,\ p, q \in G$, in the objective function of (P2), and we compute their corresponding optimal solution (ω_1, β_1).
    - Step 1.2.b2: From Lemma 2.1, we have $\theta^{pq} = \frac{\varrho_*^{12}}{\varrho_*^{pq}}$. We generate 100 uniform random values z^{pq} in $(0, 1)$ for any $p \neq q,\ p, q \in G$, and take $\theta^{pq} = \frac{z^{12}}{z^{pq}}$. By using Corollary 2.5 with (ω_1, β_1), we obtain 100 Pareto-optimal solutions of **PM**.
- Step 1.3: We calculate the testing accuracy set based on the solutions that we get from Step 1.2 for each of these multi-objective approaches.
- Step 1.4: From the sets of testing classification accuracies, we calculate the corresponding hypervolume and epsilon indicator values for each multi-objective method.

Step 2: We calculate a statistical summary for these epsilon and hypervolume indicators.

Notice that to obtain an indicator for **PM**, we only need to solve the single-objective SVM (P2) once, while for MS2, SM-OA or SM-OAO we have to solve 100 SOCPs and 50 QPs, and not all of them are guaranteed to provide a solution. Additionally, with **PM** we get approximation sets, each composed of exactly 100 testing classification accuracy vectors. So we can see that **PM** gives us a richer approximation in a shorter time, compared with MS2, SM-OA and SM-OAO.

The following boxplots (Figure 2.2 to Figure 2.9) and (Table 2.3 to Table 2.10) show the experimental results and statistical information (mean, variance, minimum, 25 percentile, median, 75 percentile and maximum) summarizing the results for the hypervolume and epsilon indicators.

We can see from these experimental results that the **PM** outperforms the other multi-objective methods. The values of the indicators obtained by **PM** are consistently among the best for the multi-objective SVMs mentioned in this chapter. For the IRIS, WINE, SEEDS, VEHICLE, CAR and GLASS data sets, **PM** has the largest mean values for both the hypervolume and

Table 2.3: Statistic information of epsilon and hypervolume indicators for IRIS data

Epsilon indicators for IRIS data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	-0.3040	0.0139	-0.6	-0.4	-0.3	-0.2	-0.1	100	59.8069
SM-OA	-0.068	0.0059	-0.3	-0.1	-0.1	0	0	100	201.7863
SM-OAO	-0.244	0.0715	-1	-0.4	-0.1	0	0	83.46	128.1809
PM	**-0.0280**	0.0025	-0.2	-0.1	0	0	0	**100**	**0.9278**
Hypervolume indicators for IRIS data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	0.7515	0.0107	0.5018	0.7031	0.7717	0.8203	0.9689	100	59.8069
SM-OA	0.9310	0.0006	0.6985	0.8975	0.8975	1	1	100	201.7863
SM-OAO	0.7919	0.0744	0	0.6985	0.9395	1	1	83.46	128.1809
PM	**0.9953**	0.0001	0.9579	0.9902	1	1	1	**100**	**0.9278**

[1] 25%= 25 percentile, 75%= 75 percentile.
[2] set size= the average approximate set size for each of the multi-objective approaches.
[3] time(s)= the average time (measured in seconds) for getting a hypervolume and epsilon indicator.

Table 2.4: Statistic information of epsilon and hyper volume indicators for WINE data

Epsilon indicators for WINE data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	-0.1290	0.0059	-0.2857	-0.2143	-0.1010	-0.0714	0	100	121.3078
SM-OA	-0.0755	0.0035	-0.2222	-0.1111	-0.0714	0	0	100	135.6761
SM-OAO	-0.1590	0.0255	-1	-0.2143	-0.1111	-0.0714	0	90.62	145.7128
PM	**-0.0576**	0.0034	-0.2143	-0.0909	-0.0714	0	0	**100**	**1.5107**
Hypervolume indicators for WINE data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	0.8661	0.0076	0.6314	0.7836	0.8800	0.9245	1	100	121.3078
SM-OA	0.9120	0.0070	0.6301	0.8802	0.9245	1	1	100	135.6761
SM-OAO	0.8415	0.0342	0	0.8096	0.8861	0.9589	1	90.62	145.7128
PM	**0.9544**	0.0038	0.7820	0.9238	0.9917	1	1	**100**	**1.5107**

Table 2.5: Statistic information of epsilon and hypervolume indicators for SEEDS data

Epsilon indicators for SEEDS data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	-0.1171	0.0031	-0.2143	-0.1429	-0.1429	-0.0714	0	97.66	185.2025
SM-OA	-0.1243	0.0041	-0.2857	-0.1429	-0.1429	-0.0714	0	100	128.2825
SM-OAO	-0.3586	0.1248	-1	-0.5714	-0.2143	-0.0714	0	96.46	156.314
PM	**-0.0771**	0.0016	-0.1429	-0.0714	-0.0714	-0.0714	0	**100**	**1.7399**
Hypervolume indicators for SEEDS data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	0.8440	0.0057	0.7162	0.7927	0.8533	0.9244	1	97.66	185.2025
SM-OA	0.8313	0.0070	0.6599	0.7868	0.8540	0.8625	1	100	128.2825
SM-OAO	0.6421	0.1295	0	0.3937	0.8038	0.9096	1	96.46	156.314
PM	**0.9647**	0.0017	0.8436	0.9309	0.9813	0.9948	1	**100**	**1.7399**

Table 2.6: Statistic information of epsilon and hypervolume indicators for VEHICLE data

Epsilon indicators for VEHICLE data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	-0.3522	0.002	-0.4524	-0.3810	-0.3488	-0.3256	-0.2619	95.72	244.0788
SM-OA	-0.3782	0.0017	-0.4762	-0.4048	-0.3765	-0.3488	-0.2857	70.18	290.3144
SM-OAO	-0.4086	0.0089	-0.7442	-0.4286	-0.3810	-0.3488	-0.2791	24.08	221.8891
PM	**-0.3497**	0.0014	-0.4286	-0.3721	-0.3488	-0.3256	-0.2857	**100**	**4.6410**
Hypervolume indicators for VEHICLE data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	0.4914	0.0026	0.3558	0.4599	0.4886	0.5257	0.6228	95.72	244.0788
SM-OA	0.5106	0.0030	0.3708	0.4771	0.5107	0.5594	0.5960	70.18	290.3144
SM-OAO	0.4490	0.0117	0.0766	0.4176	0.4685	0.5116	0.6264	24.08	221.8891
PM	**0.6220**	0.0018	0.5365	0.5970	0.6169	0.6518	0.7181	**100**	**4.6410**

Table 2.7: Statistic information of epsilon and hypervolume indicators for CAR data

Epsilon indicators for CAR data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	-0.3215	0.0103	-0.6923	-0.3846	-0.3077	-0.25	-0.1538	72.44	340.689
SM-OA	-0.8154	0.0362	-1	-1	-0.8462	-0.6154	-0.4615	92.68	392.7254
SM-OAO	-0.2985	0.0080	-0.5385	-0.3846	-0.3077	-0.2308	-0.1447	34.84	521.4667
PM	**-0.1307**	0.0005	-0.1842	-0.1488	-0.1316	-0.1184	-0.0789	**100**	**2.7537**
Hypervolume indicators for CAR data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	0.5040	0.0109	0.2292	0.4229	0.5154	0.5679	0.7368	72.44	340.689
SM-OA	0.1051	0.0120	0	0	0.0883	0.2119	0.3217	92.68	392.7254
SM-OAO	0.5442	0.0107	0.3404	0.4722	0.5395	0.6230	0.7881	34.84	521.4667
PM	**0.9309**	0.0006	0.8196	0.9226	0.9314	0.9459	0.9684	**100**	**2.7537**

Table 2.8: Statistic information of epsilon and hypervolume indicators for SCC data

Epsilon indicators for GLASS data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	-1	0	-1	-1	-1	-1	-1	73.22	139.3754
SM-OA	-1	0	-1	-1	-1	-1	-1	95.48	246.4926
SM-OAO	-1	0	-1	-1	-1	-1	-1	80.78	248.0824
PM	**-0.8344**	0.0245	-1	-1	-0.8571	-0.6667	-0.5714	**100**	**3.3367**
Hypervolume indicators for GLASS data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	0	0	0	0	0	0	0	73.22	139.3754
SM-OA	0	0	0	0	0	0	0	95.48	246.4926
SM-OAO	0	0	0	0	0	0	0	80.78	248.0824
PM	**0.0354**	0.0027	0	0	0.0151	0.0496	0.2247	**100**	**3.3367**

Figure 2.2: Boxplots of epsilon and hypervolume indicators for IRIS data

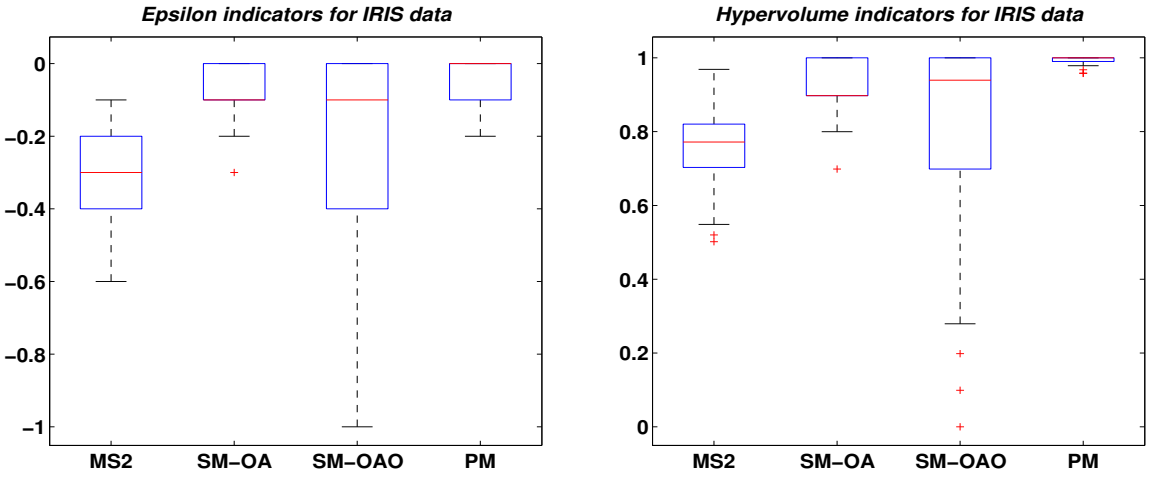
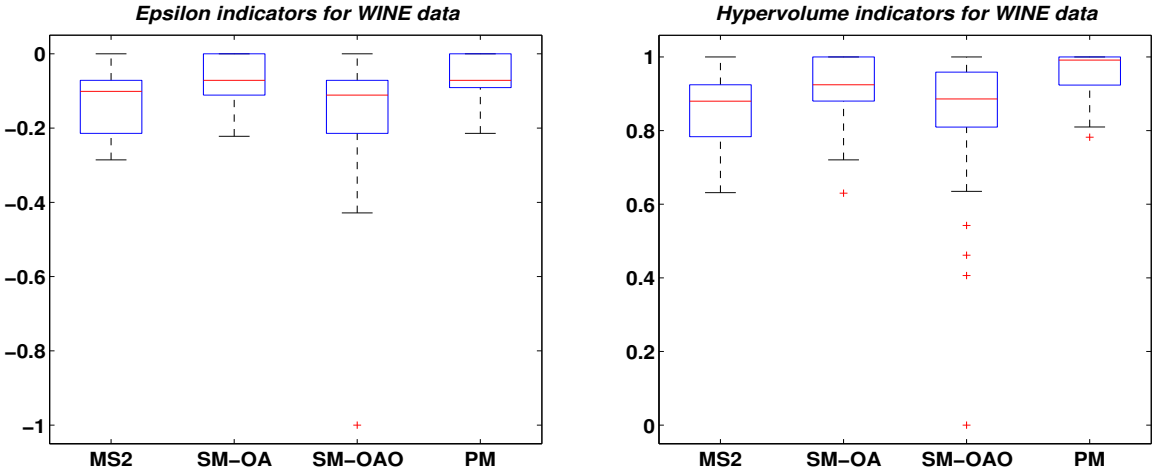


Figure 2.3: Boxplots of epsilon and hypervolume indicators for WINE data



epsilon indicators. For the SCC and CTG datasets, **PM** shows comparable performance with respect to the other three multi-objective approaches considered in this chapter. Indeed for SCC and CTG, **PM** has the second largest values of the indicators among these multi-objective methods, and these values are very close to the largest ones.

Evaluating the performances with respect to the training times and the numbers of Pareto-optimal solutions computed within those times, we can see that **PM** always outperforms the other three multi-objective methods. Indeed **PM** is at least 60 times quicker than MS2, SM-OA and SM-OAO. Moreover **PM** always obtains the largest approximation sets. As a summary, **PM** gives us more options in a shorter time compared with MS2, SM-OA and SM-OAO.

2.4 Conclusions

We have proposed a new multi-objective method (**PM**) for multi-class classification. This method is an extension of the bi-objective SVM method described in [Carrizosa and Martin-Barragan, 2006]. From the experimental results in Section 2.3, we can see that the performance of **PM** can be advantageously compared to that of the other multi-class SVMs mentioned in this chapter. In general, **PM** provides the highest classification accuracies with least training time among the multi-objective methods, and its performance is also comparable to that of the single-objective methods. **PM** provides us with a good approximation to the Pareto frontier while the single-objective methods need to conduct a large number of computations to offer us

Figure 2.4: Boxplots of epsilon and hypervolume indicators for SEEDS data

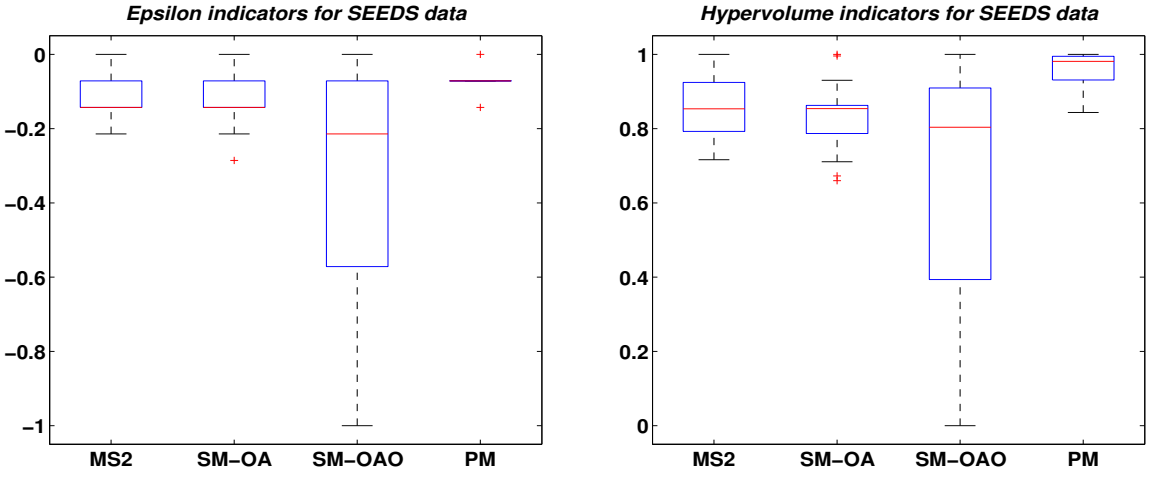


Figure 2.5: Boxplots of epsilon and hypervolume indicators for VEHICLE data

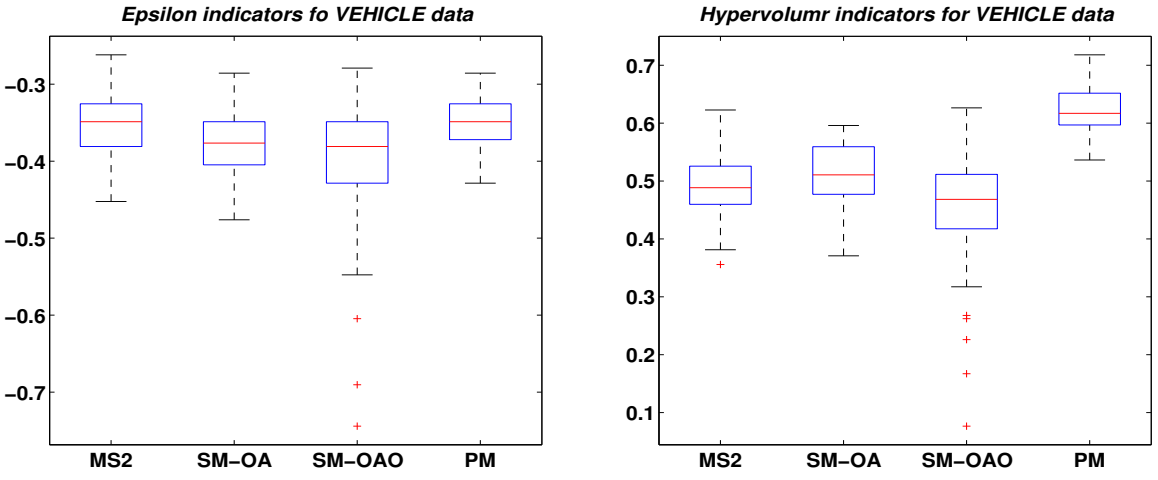


Figure 2.6: Boxplots of epsilon and hypervolume indicators for CAR data

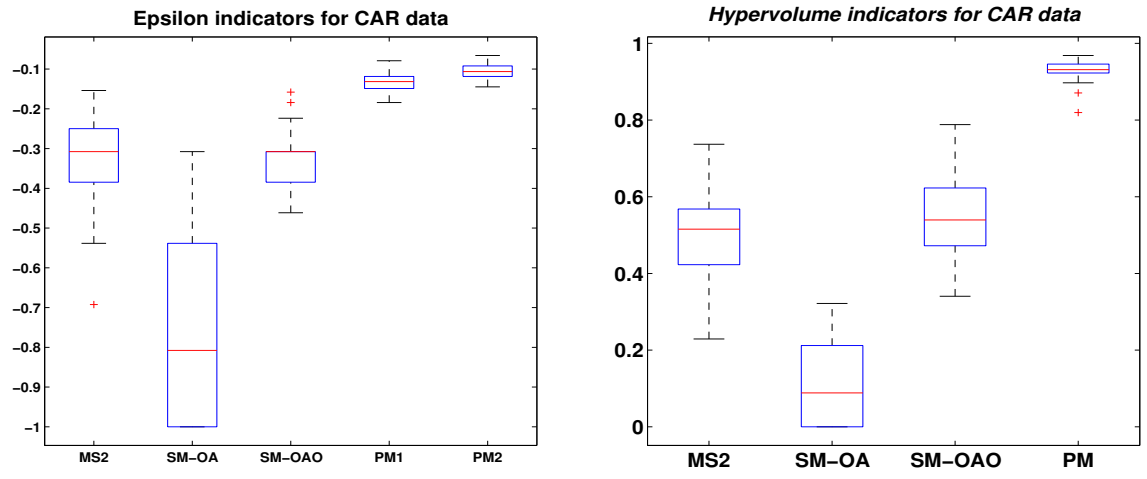


Figure 2.7: Boxplots of epsilon and hypervolume indicators for GLASS data

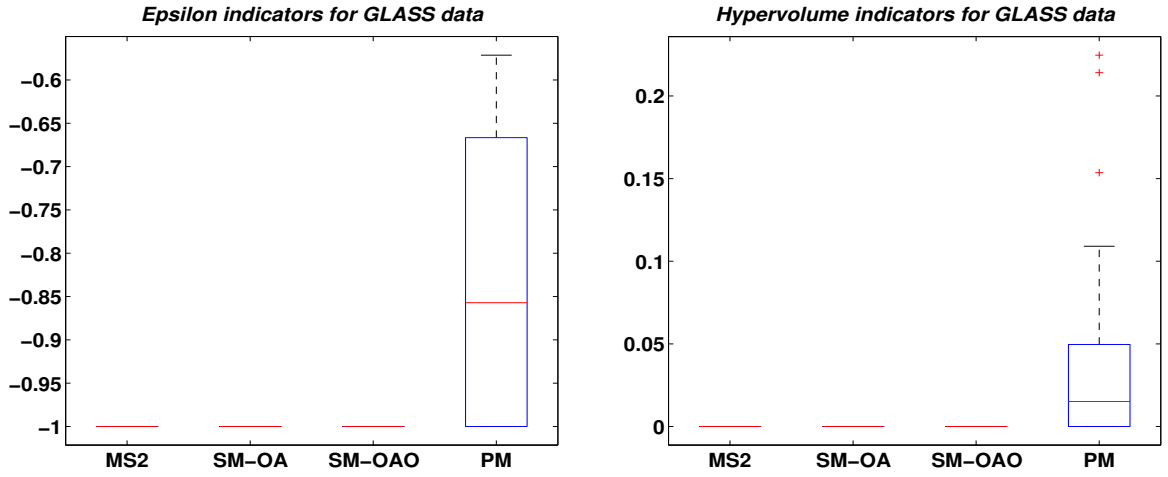


Table 2.9: Statistic information of epsilon and hyper volume indicators for SCC data

Epsilon indicators for SCC data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	**-0.018**	0.0006	-0.05	-0.05	0	0	0	74.14	285.3247
SM-OA	-0.136	0.0025	-0.25	-0.2	-0.15	-0.1	-0.05	96.66	182.1686
SM-OAO	-0.07	0.0018	-0.25	-0.1	-0.05	-0.05	0	16.76	395.1692
PM	*-0.032*	0.0006	-0.05	-0.05	-0.05	0	0	**100**	**2.1981**
Hypervolume indicators for SCC data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	0.9845	0.0005	0.9473	0.9534	1	1	1	74.14	285.3247
SM-OA	0.7570	0.0063	0.6093	0.7142	0.7662	0.8087	0.8993	96.66	182.1686
SM-OAO	0.9078	0.0043	0.7085	0.8891	0.9039	0.9494	1	16.76	395.1692
PM	**0.9981**	0.0000	0.9875	0.9977	0.9982	1	1	**100**	**2.1981**

Figure 2.8: Boxplots of epsilon and hypervolume indicators for SCC data

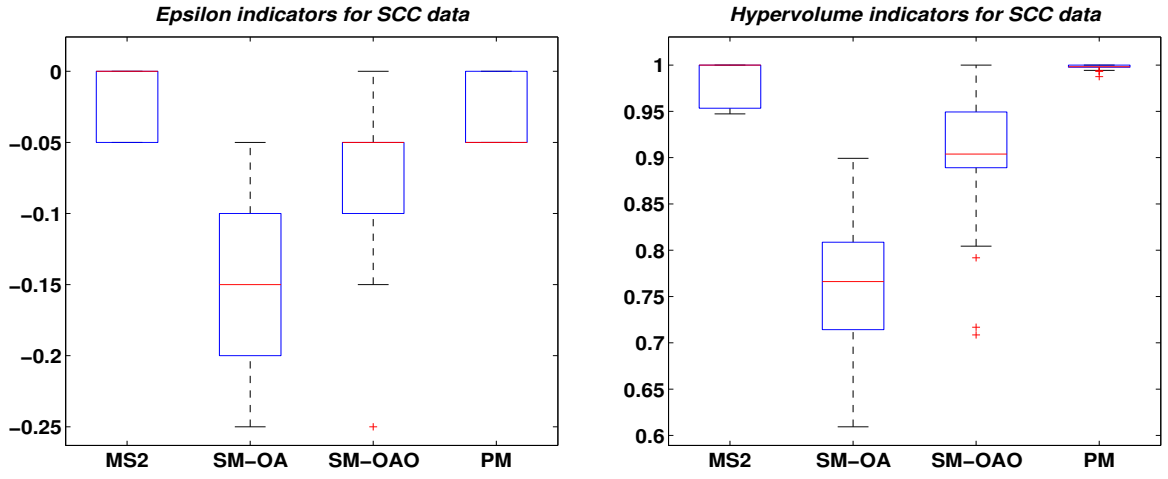


Figure 2.9: Boxplots of epsilon and hypervolume indicators for CTG data

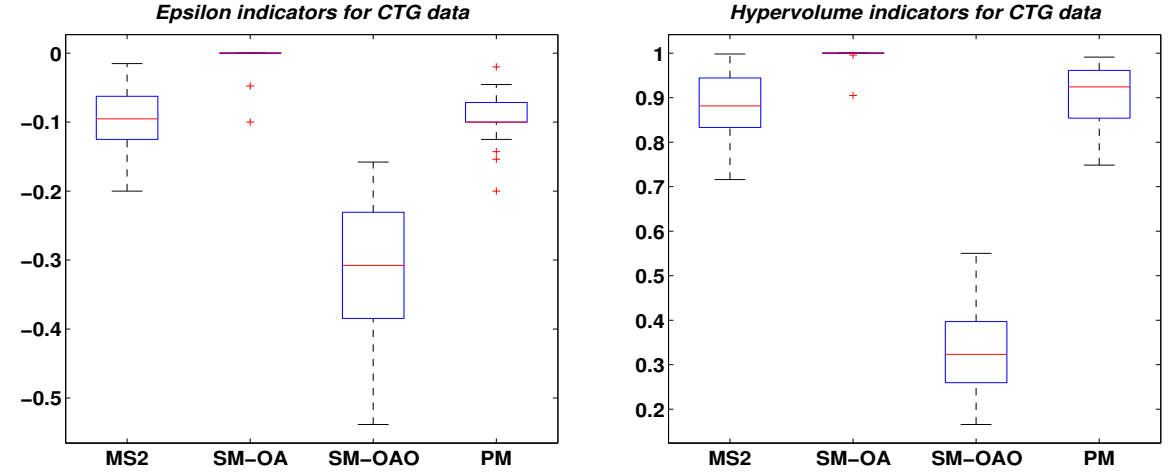


Table 2.10: Statistic information of epsilon and hypervolume indicators for CTG data

Epsilon indicators for CTG data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	-0.0934	0.0018	-0.2	-0.125	-0.0952	-0.0625	-0.0152	17.5641	9710.4
SM-OA	**-0.005**	0.0004	-0.1	0	0	0	0	82.9	1105.2
SM-OAO	-0.3131	0.0095	-0.5385	-0.3846	-0.3077	-0.2308	-0.1579	6.46	2496
PM	*-0.0936*	0.0013	-0.2	-0.1	-0.1	-0.0714	-0.02	**100**	**19.1136**
Hypervolume indicators for CTG data set									
Method	mean	variance	min	25%	median	75%	max	set size	time(s)
MS2	0.8783	0.005	0.7159	0.8332	0.8816	0.9443	0.9982	17.5641	9710.4
SM-OA	**0.9961**	0.0004	0.9050	1	1	1	1	82.9	1105.2
SM-OAO	0.3318	0.01	0.1657	0.2596	0.3234	0.3969	0.5502	6.46	2496
PM	*0.9030*	0.0048	0.7484	0.8540	0.9244	0.9614	0.9912	**100**	**19.1136**

a similar number of options to choose from. Moreover from the values of the indicators, we can also see that **PM** outperforms the other three multi-objective methods (MS2, SM-OA and SM-OAO), as it always gives us Pareto frontiers with high out-of-sample quality compared to these other multi-objective methods.

From a theoretical point of view, **PM** is also an efficient and effective method. From Corollary 2.5, **PM** provides us with Pareto-optimal solutions, while the other multi-objective approaches are only able to offer us weakly Pareto-optimal solutions. Furthermore, the Pareto-optimal solutions obtained from **PM** can be computed by solving one quadratic problem (P2). From Corollary 2.2, (P2) can be decomposed into several binary problems problem (2.11). This property significantly reduces the computational cost when the problems of interest have a large number of classes.

In summary, both from a theoretical and from a computational point of view, **PM** is an efficient method compared with MS2, SM-OA and SM-OAO. Besides, **PM**'s performance is comparable to that of the single-objective SVMs, while being able to provide not just one Pareto-optimal solution, but a very detailed approximation to the set of all the Pareto-optimal solutions.

Chapter 3

A Partial Parametric Path Algorithm for Multi-class Classification

3.1 Overview

In the classical binary SVM setting for the nonlinearly separable case, the classification problem has two objectives: avoid overfitting and limit any classification errors. This is usually modeled by combining both objectives using a parameter C, $1/2\|\omega\|^2 + CF(\sum_{i\in I}\xi_i)$, see [Cortes and Vapnik, 1995].This parameter can be seen to represent a trade-off between training and testing misclassification costs. The aim is to construct a suitable classifier which has high classification ability for the whole instance population. Attaining this aim requires the choice of a proper value for parameter C.

Selecting a default value for C may not provide acceptable results, as the optimal values from the SVM have been shown to depend critically on the choice of the value of C [Chapelle et al., 2002, Friedrichs and Igel, 2005, Hastie et al., 2004]. Usually, a grid search is used to find the appropriate value of this parameter [Friedrichs and Igel, 2005]. This approach is time-consuming, especially when we have big datasets to deal with. In [Chapelle et al., 2002], they treat C as a kernel parameter and minimize estimates of generalization errors by gradient descent. This approach depends on the differentiability of the estimates and still needs to solve the optimization problem (a quadratic program) several times.[Hastie et al., 2004] explores the entire path of binary SVMs based on the fact that the corresponding Lagrange multipliers are piecewise-linear in C. They achieved very large savings in computational costs when constructing multiple classifiers for a set of C values.

Notice that the classic binary SVM doesn't take into account any a priori information (such as skewed class distributions, different misclassification costs). However, this information may

be critical. For example in medical diagnosis, the difference between the classification costs of misclassifying a healthy person as ill and a diseased patient as healthy is large and can't be ignored. To take into account these information, [Bach, 2006, Veropoulos et al., 1999] use two different parameters C^+ and C^- associated to different classes. Also, [Bach, 2006] extends the path algorithm when considering asymmetric misclassification costs. In [Karasuyama et al., 2012], differences between the weights of instances are taken into account and a corresponding path algorithm is proposed. Their numerical experiments show that these proposed path algorithms significantly reduce the computational cost to find proper parameter values for the binary SVMs.

As in real life we usually have more than two classes, the efficient two-class SVM approach has been extended for multi-class classifications. In [Weston and Watkins, 1998], they propose a single-objective SVM to handle all the classes simultaneously. However, they consider all misclassification costs to be the same, and they use no a priori information related to the classes' distributions. A direct way to overcome this drawback is to assign different weights to the penalty terms for different misclassification errors in the objective function. Another way is to use multi-objective methods as we have mentioned in Chapter 2. As we discussed before, the first proposal, based on a direct assignment of weight values, is not efficient because of the large computational costs required to choose suitable weights.

L.Wang and X.Shen [Lifeng Wang, 2006] have proposed a path algorithm for multi-class classification problems based on the L_1 norm. It takes advantage of the property that their optimal solutions are piecewise-linear on a tuning parameter s, which controls the sum of the L_1 norms of all the slope vectors ω_c. They reconstruct the entire optimal path based on finding appropriate features characterizing changes in the active sets. To identify the joints and get the solutions, they need to construct sets of linear equations based on the derivatives of the slopes with respect to s. Our proposal shares its basic motivation with this one, but it aims to take advantage of the simplicity of L_2 optimization problems, and to take into account differences among classification costs. We introduce a partial-parametric-path algorithm (**PPPA**) for multi-class classification inspired by the previous one and by [Bach, 2006, Hastie et al., 2004]. The partial path begins with a starting solution obtained from a multi-class SVM problem, and it is extended to different values of the weight parameters, while checking if they are acceptable.

In this chapter, we provide a general framework for the application of **PPPA**. If we have a sufficiently good classification performance at the starting point (a default set of values for the parameters), we don't need to apply **PPPA**. If the initial classification performance is not acceptable, we take advantage of the piecewise linearity of the optimal solutions to obtain efficient representations of the solution paths for alternative values of the parameters. By controlling the changes in the active sets, we construct partial solution paths along some chosen parameter directions. To systematically explore the whole parameter space, we combine **PPPA** with a variable neighborhood search method (VNS). When using **PPPA**, we only need to solve one quadratic program to get our starting solution. All other solutions are obtained by solving systems of linear equations. Thus, **PPPA** is computationally efficient. From our experiments

in Section 3.6, we have also verified that **PPPA** is robust, as it provides the same solutions as the ones obtained from the corresponding quadratic programming problems, in nearly all cases.

This chapter is organized as follows: In Section 3.2, we present a single-objective multi-class SVM which takes into account differences in misclassification costs. In Section 3.3, we characterize the piecewise linear nature of the optimal solutions of the quadratic programs of interest. The components of the partial solution path are presented in Section 3.4. The combination of **PPPA** and VNS is introduced in Section 3.5. In Section 3.6, we describe and comment a set of experimental results showing that **PPPA** is efficient and reliable for multi-class classification problems. Finally, conclusions are presented in Section 3.7.

3.2 Our reference multi-class support vector machine

For simplicity, we still just consider linear classifiers as nonlinear ones can be seen as linear in a higher dimensional space. For multi-class classifications, the most commonly used single-objective methods are the all-together, one-against-all and one-against-one methods, [Hsu and Lin, 2002]. The all-together and one-against-all methods are based on constructing m classifiers. Specifically, the all-together method maximizes the sum of all the functional margins and minimizes the penalty variables simultaneously within a single quadratic programming. Each of the functional margins is constructed by considering all instances of one class vs those of all the remaining classes; a similar approach is used in one-against-all method. One-against-all methods solve m quadratic programs to obtain the m classifiers. In all these cases, the presence of asymmetries in the misclassification costs, for example, may have a significant effect on the accuracies of the classifiers.[1] A one-against-one method constructs $m(m-1)/2$ classifiers and each of them are obtained from a binary SVM which just considers a pair of classes. In [Hsu and Lin, 2002], experiments show that in general a one-against-all method does not achieve accuracies as high as a one-against-one method. In this chapter we prefer to construct $m(m-1)/2$ classifiers and maximize all the pairwise functional margins. Of course, we could use the path-algorithm (introduced in [Bach, 2006]) on each of the binary SVMs from the one-against-one method to find a satisfactory choice of parameters. But when m is large, this would require tracking a very large number of paths, with very high associated computational costs. In order to take advantage of the high accuracy of a one-against-one method and the compactness of an all-together method, we have chosen to construct a single-objective multi-class SVM maximizing all the pairwise functional margins and minimizing all the penalties at the same time.

We construct our classifiers as follows:

[1] A one-against-all classifier is constructed from m binary SVMs. Each of these binary SVMs considers only one class as the positive class and all the remaining classes as the negative class. So we can see that the difference between the sizes of positive and negative classes can be quite large. The classification accuracies would be affected because the classical binary SVM gives the same weight to the penalties of the positive and negative classes misclassification errors.

- The classifier (discriminate hyperplane) separating the training data from class p and class q, is given by:

$$L^{pq} : f^{pq}(x) = (\omega^{pq})^T x + \beta^p - \beta^q = 0, q > p, p, q \in G.$$

Ideally, we would like to have all class p objects lying above hyperplane $L^{pq}, q \neq p, p, q \in G$, and all class q objects lying below L^{pq}. If there exist hyperplanes such that the training objects satisfy this ideal situation, we say that the training objects are linearly separable. But usually we have linearly nonseparable data, and we need to take into account both functional margins and misclassification errors. So, by maximizing all the functional margins of the $m(m-1)/2$ classifiers and minimizing the misclassification penalties, we construct a single-objective SVM,

$$\begin{aligned} \min_{\omega,\beta,\xi} \quad & \frac{1}{2} \sum_{q>p,p,q\in G} (\omega^{pq})^T \omega^{pq} + \sum_{q\neq p,p,q\in G} \sum_{i\in I_p} t_0^{pq} \xi_i^{pq}, \\ s.t. \quad & (\omega^{pq})^T x_i + \beta^p - \beta^q + \xi_i^{pq} \geq 1, i \in I_p, q > p, p, q \in G, \\ & -(\omega^{pq})^T x_i - \beta^p + \beta^q + \xi_i^{qp} \geq 1, i \in I_q, q > p, p, q \in G, \\ & \xi_i^{pq} \geq 0, i \in I_p, q \neq p, p, q \in G, \end{aligned} \tag{3.1}$$

where $\omega = (\omega^{12^T}, \omega^{13^T}, \cdots, \omega^{(m-1)m^T})^T \in \mathbb{R}^{m(m-1)l/2}$, $\beta = (\beta^1, \beta^2, \cdots, \beta^m)^T \in \mathbb{R}^m$, $\xi \in \mathbb{R}^{(m-1)n}$ collecting $\xi_i^{pq}, i \in I_p, q \neq p, p, q \in G$ and t_0^{pq} denotes the weights for each of the penalties. Misclassification penalties ξ_x^{pq} are introduced to avoid overfitting and to guarantee the existence of solutions for problem (3.1) in the linearly nonseparable case. To take into account the possible differences in misclassification costs, we introduce different weights $t_0^{pq}, q \neq p, p, q \in G$ for each misclassification penalties associated to different classes and classifiers.

Note that problem (3.1) is a quadratic program, whose solution provides the information to define the corresponding classifiers. A difficulty is the choice of acceptable values for the weights t_0^{pq}, that is, values that yield classifiers which have high accuracies and low misclassification costs.

This chapter focuses on the introduction of efficient procedures to determine acceptable sets of weights. Our approach is based on a two-step approach:

- We select an initial set of weights, and a "search direction" on the space of these weights.
- We find the optimal combination of weights on this one-dimensional space by building a partial path from the optimal solution of problem (3.1) as the starting point, and conducting a search on these solutions.
- Finally, we modify the search directions using a variable neighborhood search method (VNS) and repeat this process until we are close enough to an acceptable solution.

To build the partial path we solve the following program as a function of C:

$$\begin{aligned}
\min_{\omega,\beta,\xi} \quad & \frac{1}{2}\sum_{q>p,p,q\in G}(\omega^{pq})^T\omega^{pq} + \sum_{q\neq p,p,q\in G}\sum_{i\in I_p} t_0^{pq}\xi_i^{pq} + C\sum_{q\neq p,p,q\in G}\sum_{i\in I_p} t_1^{pq}\xi_i^{pq}, \\
s.t. \quad & (\omega^{pq})^T x_i + \beta^p - \beta^q + \xi_i^{pq} \geq 1, i \in I_p, q > p, p, q \in G, \\
& -(\omega^{pq})^T x_i - \beta^p + \beta^q + \xi_i^{qp} \geq 1, i \in I_q, q > p, p, q \in G, \\
& \xi_i^{pq} \geq 0, i \in I_p, q \neq p, p, q \in G,
\end{aligned} \tag{3.2}$$

here $t_1^{pq}, q \neq p, p, q \in G$ denotes the direction along which we construct the partial path. After identifying appropriate parameter values and building the corresponding classifiers, we use the majority voting to define our classification rule.

3.3 Optimal classifiers are piecewise affine functions of C

In this Section, we prove that the solutions of problem (3.2)are piecewise affine wrt C. This is the basic property on which the efficiency of our proposal rests. We also introduce simple characterizations for the solutions of interest.

As the classifiers $L^{pq} : (\omega^{pq})^T x + \beta^p - \beta^q = 0, q > p, p, q \in G$ depend only on the differences $\beta^p - \beta^q$ (instead of the values β^p), without loss of generality we will set $\beta^1 = 0$ in all that follows.

We will use a slightly modified notation, to simplify the representation of the optimal solutions of problem (3.2). In particular, we will work with the observations projected onto a higher dimensional space. Let

$$X_x^{pq} = \left(\delta_{x,pq}^{12}, \delta_{x,pq}^{13}, \cdots, \delta_{x,pq}^{(m-1)m}\right), \quad \text{where} \quad \delta_{x,pq}^{ij} = \begin{cases} x, & \text{if } (i,j) = (p,q), \\ -x, & \text{if } (i,j) = (q,p), \\ 0, & \text{otherwise.} \end{cases}$$

We can write $(\omega^{pq})^T x + \beta^p - \beta^q = (\omega)^T X_x^{pq} + \beta^p - \beta^q$. Let X^{pq} denote a matrix with row vectors equal to $X_x^{pq}, x \in I_p$, while ξ^{pq} denotes a vector containing the values $\xi_x^{pq}, x \in I_p$. Define

$$X = \begin{pmatrix} X^{12} \\ X^{21} \\ \vdots \\ X^{(m-1)m} \\ X^{m(m-1)} \end{pmatrix}, \xi = \begin{pmatrix} \xi^{12} \\ \xi^{21} \\ \vdots \\ \xi^{(m-1)m} \\ \xi^{m(m-1)} \end{pmatrix} \text{ and } t_k = \begin{pmatrix} t_k^{12}\vec{1}_{n^1\times 1} \\ t_k^{21}\vec{1}_{n^2\times 1} \\ \vdots \\ t_k^{(m-1)m}\vec{1}_{n^{m-1}\times 1} \\ t_k^{m(m-1)}\vec{1}_{n^m\times 1} \end{pmatrix}, k = 0, 1,$$

where $n^p, p \in G$ is the number of class p training instances. Then, we can rewrite problem (3.2)as follows:

$$\begin{aligned} \min_{\omega,\beta,\xi} \quad & \frac{1}{2}\|\omega\|^2 + (t_0)^T\xi + C(t_1)^T\xi, \\ \text{s.t.} \quad & X\omega + R\beta + \xi \geq \vec{1}_{(m-1)n\times 1}, \\ & \xi \geq \vec{0}_{(m-1)n\times 1}, \end{aligned} \tag{3.3}$$

where $\omega \in \mathbb{R}^{(m-1)ml/2}$, $\beta = (\beta^2, \beta^3, \cdots, \beta^m)^T \in \mathbb{R}^{m-1}$, $\xi \in \mathbb{R}^{(m-1)n}$ and R is the coefficient matrix of the variables β in problem (3.2).

As the objective function of problem (3.3) is quadratic (positive semidefinite) and the constraints are affine functions, the corresponding KKT conditions are necessary and sufficient for optimality. These KKT conditions for problem (3.3) are:

$$\omega = X^T\lambda, \tag{3.4}$$
$$R^T\lambda = \vec{0}_{(m-1)\times 1}, \tag{3.5}$$
$$\lambda + \mu = t_0 + Ct_1, \tag{3.6}$$
$$X\omega + R\beta + \xi \geq \vec{1}_{(m-1)n\times 1}, \tag{3.7}$$
$$\xi \geq \vec{0}_{(m-1)n\times 1}, \tag{3.8}$$
$$\lambda \geq \vec{0}_{(m-1)n\times 1}, \tag{3.9}$$
$$\mu \geq \vec{0}_{(m-1)n\times 1}, \tag{3.10}$$
$$\lambda^T(\vec{1}_{(m-1)n\times 1} - X\omega - R\beta - \xi) = 0, \tag{3.11}$$
$$\mu^T\xi = 0, \tag{3.12}$$

where λ denotes the multipliers of $X\omega + R\beta + \xi \geq \vec{1}_{(m-1)n\times 1}$ and μ denotes the multipliers of $\xi \geq \vec{0}_{(m-1)n\times 1}$.

We define the following relevant active sets:

- The indices of the above-margin objects: $A = \{i \mid X_i\omega + R_i\beta > 1\}$,
- The indices of the below-margin objects: $B = \{i \mid X_i\omega + R_i\beta < 1\}$,
- The indices of the on-margin objects: $O = \{i \mid X_i\omega + R_i\beta = 1\}$,

where X_i and R_i denotes the i-th row of X and R respectively. As each constraint is associated to one object, these definitions allow us to separate the objects into three disjoint sets.

Based on the preceding KKT conditions, we refine the preceding definitions as follows:

$$\begin{aligned} & i \in A, \text{ if and only if } \lambda_i = 0, \\ & i \in B, \text{ if and only if } \lambda_i = t_{0i} + Ct_{1i}, \\ & i \in O, \text{ if and only if } \ 0 < \lambda_i < t_{0i} + Ct_{1i}. \end{aligned}$$

Let X_A denote the sub-matrix of X collecting all rows X_i with $i \in A$. Similarly, we define X_B, X_O, R_A, R_B, and R_O. Also, let λ_O denote the sub-vector of λ corresponding to the components λ_i with $i \in O$. Similarly, we introduce $\lambda_A, \lambda_B, t_{0A}, t_{0B}, t_{0O}, t_{1A}, t_{1B}$ and t_{1O}. We have

$$\begin{aligned}
&\omega = X_O^T \lambda_O + X_B^T t_{0B} + C X_B^T t_{1B}, \\
&R_O^T \lambda_O + R_B^T t_{0B} + C R_B^T t_{1B} = \vec{0}_{(m-1)\times 1}, \\
&X_O \omega + R_O \beta = \vec{1}_{n_O \times 1},
\end{aligned}$$

where n_O denotes the size of the active set O.

From these definitions and the optimality conditions, the solutions of problem (3.3) can be computed by solving the system

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & X_O^T \\ \mathbf{0} & \mathbf{0} & R_O^T \\ X_O & R_O & \mathbf{0} \end{pmatrix} \begin{pmatrix} \omega \\ \beta \\ -\lambda_O \end{pmatrix} = \begin{pmatrix} X_B^T t_{0B} \\ R_B^T t_{0B} \\ \vec{1}_{n_O \times 1} \end{pmatrix} + C \begin{pmatrix} X_B^T t_{1B} \\ R_B^T t_{1B} \\ \vec{0}_{n_O \times 1} \end{pmatrix}, \tag{3.13}$$

where X_O is a $n_O \times \frac{m(m-1)l}{2}$ matrix and R_O is a $n_O \times (m-1)$ matrix.

We will make use of the following auxiliary result characterizing some properties of the coefficient matrix for system (3.13).

Lemma 3.1. *Given active sets O, A and B, the necessary and sufficient conditions for the coefficient matrix of (3.13) to be nonsingular is that the column vectors of R_O are linearly independent and the row vectors of $\begin{pmatrix} X_O & R_O \end{pmatrix}$ are also linearly independent.*

Proof. Consider an auxiliary system having the same coefficient matrix as system (3.13),

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & X_O^T \\ \mathbf{0} & \mathbf{0} & R_O^T \\ X_O & R_O & \mathbf{0} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \vec{0}. \tag{3.14}$$

We will use it to study the singularity of the coefficient matrix of (3.13), based on the properties of the solutions of problem (3.14).

- Necessity: Suppose the coefficient matrix of (3.13) and problem (3.14)is nonsingular, then the unique solution of problem (3.14) is $u = \vec{0}$.

  If the column vectors of R_O are linearly dependent, there exists $u_2 \neq \vec{0}$ such that $R_O u_2 = \vec{0}$. The coefficient matrix of (3.13) cannot be invertible in this case, as we can choose $u_1 = \vec{0}$, $u_3 = \vec{0}$ and obtain a nonzero vector in the null space of this matrix, contradicting our assumption. It follows that if the coefficient matrix of (3.13) is nonsingular, the column vectors of R_O must be linearly independent.

Analogously, if the row vectors of $\begin{pmatrix} X_O & R_O \end{pmatrix}$ are linearly dependent, there exists $u_3 \neq \vec{0}$ such that

$$\begin{pmatrix} X_O^T \\ R_O^T \end{pmatrix} u_3 = \vec{0}.$$

Again, in this case we can choose $u_1 = \vec{0}$, $u_2 = \vec{0}$ and obtain a nonzero vector in the null space of this matrix. This contradicts our assumption, implying that we must have linearly independent row vectors for $(X_O \; R_O)$.

- Sufficiency: Assume that the column vectors of R_O and the row vectors of $\begin{pmatrix} X_O & R_O \end{pmatrix}$ are linearly independent. This implies that $M \triangleq X_O X_O^T + R_O R_O^T$ and $R_O^T R_O$ are invertible, and M is positive definite.

  From problem (3.14) we have

  $$\begin{aligned}
  &R_O^T u_3 = \vec{0} \Rightarrow R_O R_O^T u_3 = \vec{0} \\
  &u_1 = -X_O^T u_3 \\
  &X_O u_1 + R_O u_2 = \vec{0} \Rightarrow R_O u_2 = X_O X_O^T u_3 = (X_O X_O^T + R_O R_O^T) u_3 \Rightarrow u_3 = M^{-1} R_O u_2 \\
  &R_O^T u_3 = R_O^T M^{-1} R_O u_2 = \vec{0}.
  \end{aligned}$$

  As M is positive definite, $R_O^T M^{-1} R_O$ is at least positive semidefinite. As $u_2^T R_O^T M^{-1} R_O u_2 = 0$, from the positive-definiteness of M we have $R_O u_2 = \vec{0}$. As R_O has full column rank, it must hold that $u_2 = \vec{0}$. As $u_3 = M^{-1} R_O u_2$ and $u1 = -X_O^T u_3$, we have $u_1 = \vec{0}$ and $u_3 = \vec{0}$. It follows that the coefficient matrix of (3.13) and problem (3.14) is nonsingular.

□

To present our existence result, we introduce the following regularity condition:

$\mathcal{C}1$: The row vectors of $\begin{pmatrix} X_O & R_O \end{pmatrix}$ are linearly independent.

The next result provides the desired characterization of the solutions.

Theorem 3.2. *Under condition $\mathcal{C}1$, the solution of (3.13) defines a piecewise-affine optimal solution of problem (3.3).*

Proof. As $\beta = (\beta^2, \beta^3, \cdots, \beta^m)^T$, the number of columns in R_O and its maximum column and row rank is $m-1$. Consider an optimal solution of problem (3.3) defined by $(\omega_0, \beta_0, \xi_0)$.

Assume that R_O does not have full rank. Then, there exists a non-zero vector u such that $R_O u = 0$. Define $\beta_* = \beta_0 + \epsilon u$, and

$$\xi_{B*}^{pq} = \vec{1} - X_B^{pq} \omega_0 - (\beta_*^p - \beta_*^q) = \xi_{B0}^{pq} - \epsilon(u^p - u^q).$$

As $R_O u = 0$ it holds that $X_O\omega_0 + R_O\beta_* = \vec{1}$ for any ϵ, and $(\omega_0, \beta_*, \xi_*)$ are feasible as long as $\epsilon_1 \leq \epsilon \leq \epsilon_2$, with

$$\epsilon_1 = \max\left\{ \max_{u^p-u^q<0, p\neq q, p,q\in G} \frac{X_B^{pq}\omega_0 + \beta_0^p - \beta_0^q - \vec{1}}{-(u^p-u^q)}, \max_{u^p-u^q<0, p\neq q, p,q\in G} \frac{X_A^{pq}\omega_0 + \beta_0^p - \beta_0^q - \vec{1}}{u^p-u^q} \right\} < 0,$$

$$\epsilon_2 = \min\left\{ \min_{u^p-u^q>0, p\neq q, p,q\in G} \frac{X_A^{pq}\omega_0 + \beta_0^p - \beta_0^q - \vec{1}}{u^p-u^q}, \min_{u^p-u^q>0, p\neq q, p,q\in G} \frac{X_B^{pq}\omega_0 + \beta_0^p - \beta_0^q - \vec{1}}{-(u^p-u^q)} \right\} > 0.$$

Select $\epsilon = \epsilon_1$ if

$$\sum_{p\neq q,p,q\in G, u^p-u^q<0} (u^q-u^p)(t_0^{pq} + Ct_1^{pq})\#B^{pq} > \sum_{p\neq q,p,q\in G, u^p-u^q>0} (u^p-u^q)(t_0^{pq} + Ct_1^{pq})\#B^{pq},$$

and $\epsilon = \epsilon_2 > 0$ otherwise. We have that

$$\begin{aligned}
\Phi_* &= \frac{1}{2}\|\omega_0\|^2 + (t_0 + Ct_1)^T\xi_*, \\
&= \Phi_0 + \epsilon\left[\sum_{p\neq q,p,q\in G, u^p-u^q<0} (u^q-u^p)(t_0^{pq} + Ct_1^{pq})\#B^{pq} \right. \\
&\quad \left. - \sum_{p\neq q,p,q\in G, u^p-u^q>0} (u^p-u^q)(t_0^{pq} + Ct_1^{pq})\#B^{pq} \right] \leq \Phi_0,
\end{aligned}$$

where $\Phi_0 = \frac{1}{2}\|\omega_0\|^2 + (t_0 + Ct_1)^T\xi_0$. As a consequence, there must exist an optimal solution including an additional support vector in O, the one defining the selected value for ϵ. This procedure can be repeated until the matrix R_O has full column rank.

As we can always find a solution such that the column vectors of R_O are linearly independent, if condition $\mathcal{C}1$ holds for active sets O, A and B, from Lemma 3.1 the inverse of the coefficient matrix of (3.13) exists, and (3.13) implies that the optimal solution $(\omega, \beta, \lambda_O)$ is an affine function of C.

Also, as $\xi_i = \max(0, 1 - X_i\omega - R_i\beta)$, $\lambda_A = \vec{0}$, $\lambda_B = t_{0B} + Ct_{1B}$ and $\mu = t_0 + Ct_1 - \lambda$, the optimal solutions $(\omega, \beta, \xi, \lambda, \mu)$ are affine with respect to C, under our conditions. Thus, under $\mathcal{C}1$ the optimal classifiers are piecewise affine functions of C. □

From this result, the coefficient matrix of (3.13) is singular only if the row vectors of $\begin{pmatrix} X_O & R_O \end{pmatrix}$ are linearly dependent. If this is the case, we could project the observations onto a higher dimensional space in which the number of on-margin observations may be smaller and the row vectors of the corresponding $\begin{pmatrix} X_O & R_O \end{pmatrix}$ matrix are linearly independent. Thus, for simplicity we will assume in what follows that the row vectors of $\begin{pmatrix} X_O & R_O \end{pmatrix}$ are linearly independent.

3.4 The partial parametric path algorithm

To be able to identify good parameter values for the weights of the misclassification errors t, we follow the procedure described in Section 3.2. The basic step in that procedure consists in the construction of a univariate path along a given parameter direction (selected using a VNS method). In this Section, we study the properties of this path and we present efficient methods to obtain it.

This procedure consists on the following steps:

- For a given starting parameter vector t_0 and a parameter direction t_1, we compute the solution to the SVM problem corresponding to $C = C_0 = 0$.
- Then we determine the largest increase in the parameter C that will not change the active sets, C_1. We obtain this value by making use of the linear structure of the optimal solutions in that interval, using Theorem Theorem 3.2.
- We update the active sets at C_k with $k = 1$ (a "joint" in the path), and we repeat the procedure for increasing k until some criterion is optimized, or a stopping criterion is satisfied. In our experiments, the criterion to optimize has been the training classification accuracy, and the termination criterion has been defined as reaching a prespecified maximum value for C.

3.4.1 Characterization of linear segments of the solution path

From Theorem 3.2, for given active sets the classifiers are affine with respect to parameter C. Define C_k as the k-th joint where there is a change in the active sets, B_k, A_k, O_k are the active sets corresponding to all values $C_k \leq C < C_{k+1}$, and λ_k, μ_k denote the corresponding multipliers.

As the assume that the row vectors of $\begin{pmatrix} X_{O_k} & R_{O_k} \end{pmatrix}$ are linearly independent, we can define

$$\begin{pmatrix} \omega_k^a \\ \beta_k^a \\ -\lambda_{k,O_k}^a \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & X_{O_k}^T \\ \mathbf{0} & \mathbf{0} & R_{O_k}^T \\ X_{O_k} & R_{O_k} & \mathbf{0} \end{pmatrix}^{-1} \begin{pmatrix} X_{B_k}^T t_{0B_k} \\ R_{B_k}^T t_{0B_k} \\ \vec{1} \end{pmatrix},$$

and

$$\begin{pmatrix} \omega_k^b \\ \beta_k^b \\ -\lambda_{k,O_k}^b \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & X_{O_k}^T \\ \mathbf{0} & \mathbf{0} & R_{O_k}^T \\ X_{O_k} & R_{O_k} & \mathbf{0} \end{pmatrix}^{-1} \begin{pmatrix} X_{B_k}^T t_{1B_k} \\ R_{B_k}^T t_{1B_k} \\ \vec{0} \end{pmatrix}.$$

Also,

$$\lambda_{k,A_k}^a = \vec{0}, \quad \lambda_{k,A_k}^b = \vec{0}, \quad \lambda_{k,B_k}^a = t_{0B_k}, \quad \lambda_{k,B_k}^b = t_{1B_k},$$

$$\begin{aligned}
&\mu_k^a = t_0 - \lambda_k^a, \quad \mu_k^b = t_1 - \lambda_k^b, \\
&\xi_{k,A_k\cup O_k}^a = \vec{0}, \quad \xi_{k,B_k}^a = \vec{1} - X_{B_k}\omega_k^a - R_{B_k}\beta_k^a, \\
&\xi_{k,A_k\cup O_k}^b = \vec{0}, \quad \xi_{k,B_k}^b = -X_{B_k}\omega_k^b - R_{B_k}\beta_k^b.
\end{aligned}$$

Then the optimal solution $(\omega, \beta, \xi, \lambda, \mu)$ of problem (3.3) with respect to C, $C_k \leq C < C_{k+1}$ is given by:

$$\begin{pmatrix} \omega \\ \beta \\ \xi \\ \lambda \\ \mu \end{pmatrix} = \begin{pmatrix} \omega_k^a \\ \beta_k^a \\ \xi_k^a \\ \lambda_k^a \\ \mu_k^a \end{pmatrix} + C \begin{pmatrix} \omega_k^b \\ \beta_k^b \\ \xi_k^b \\ \lambda_k^b \\ \mu_k^b \end{pmatrix}. \tag{3.15}$$

3.4.2 Finding the joint values C_k

Between joints, the active sets do not change and the optimal solutions of problem (3.3) are affine functions of C. But as we increase the value of C, the solution path problem (3.15) reaches values that may not satisfy some of the optimality conditions. In this Section, we describe how to obtain efficiently the value of the next joint (the value of C), where the active sets change.

As the value of C changes from the preceding joint, at some point it will become necessary to adjust the active sets. We can classify these adjustments into four cases:

- Case 1: Some above-margin object changes to become an on-margin one;
- Case 2: Some below-margin object changes to become an on-margin one;
- Case 3: Some on-margin object changes to become an above-margin one;
- Case 4: Some on-margin object changes to become a below-margin one.

We now provide the characterization of the joint values corresponding to each of the four cases, for $C \geq C_k$,

- For the first case, we check $X_i\omega + R_i\beta = 1$, for $i \in A_k$. We have

$$X_i(\omega_k^a + C\omega_k^b) + R_i(\beta_k^a + C\beta_k^b) = 1 \Rightarrow C = \frac{1 - (X_i\omega_k^a + R_i\beta_k^a)}{X_i\omega_k^b + R_i\beta_k^b}, i \in A_k.$$

  As $X_i(\omega_k^a + C_k\omega_k^b) + R_i(\beta_k^a + C_k\beta_k^b) > 1$, we only need to check $X_i\omega + R_i\beta = 1$, for $i \in A_k$ and $X_i\omega_k^b + R_i\beta_k^b < 0$. We define

$$C_{k+1}^1 = \min\left\{ \frac{1 - (X_i\omega_k^a + R_i\beta_k^a)}{X_i\omega_k^b + R_i\beta_k^b} | i \in A_k \text{ and } X_i\omega_k^b + R_i\beta_k^b < 0 \right\}.$$

- For the second case, we check $\xi_i = \xi^a_{k,i} + C\xi^b_{k,i} = 0$ for $i \in B_k$. As $\xi^a_{k,i} + C_k\xi^b_{k,i} > 0$ for $i \in B_k$, we only need to check $\xi^a_{k,i} + C\xi^b_{k,i} = 0$ for $i \in B_k$ and $\xi^b_{k,i} < 0$. We define

$$C^2_{k+1} = \min\left\{-\frac{\xi^a_{k,i}}{\xi^b_{k,i}} | i \in B_k \text{ and } \xi^b_{k,i} < 0\right\}.$$

- For the third case, we check $\lambda_i = \lambda^a_{k,i} + C\lambda^b_{k,i} = 0$, for $i \in O_k$. As $\lambda^a_{k,i} + C_k\lambda^b_{k,i} > 0$ for $i \in O_k$, we only need to check $\lambda^a_{k,i} + C\lambda^b_{k,i} = 0$, for $i \in O_k$ and $\lambda^b_{k,i} < 0$. We define

$$C^3_{k+1} = \min\left\{-\frac{\lambda^a_{k,i}}{\lambda^b_{k,i}} | i \in O_k \text{ and } \lambda^b_{k,i} < 0\right\}.$$

- For the last case, we check $\mu_i = \mu^a_{k,i} + C\mu^b_{k,i} = 0$ for $i \in O_k$. As $\mu^a_{k,i} + C_k\mu^b_{k,i} > 0$ for $i \in O_k$, we only need to check $\mu^a_{k,i} + C\mu^b_{k,i} = 0$ for $i \in O_k$ and $\mu^b_{k,i} < 0$. We define

$$C^4_{k+1} = \min\left\{-\frac{\mu^a_{k,i}}{\mu^b_{k,i}} | i \in O_k \text{ and } \mu^b_{k,i} < 0\right\}.$$

The next joint value is defined as:

$$C_{k+1} = \min\{C^1_{k+1}, C^2_{k+1}, C^3_{k+1}, C^4_{k+1}\}.$$

At C_{k+1}, we can update the new values $(\omega_{k+1}, \beta_{k+1}, \xi_{O_{k+1}})$ by observing the changes in the active sets and solving equations (3.13) for the updated active sets. Then we repeat the procedure with the updated active sets to find the next joint.

It would be possible to build an entire path starting from t_0 close to $\vec{0}$. But we have found that this is not useful in practice in many cases, as problem (3.1) would return $\omega^{pq} = \vec{0}, q > p, p, q \in G$ for this value, which is not a very useful choice for data classification. A reasonable initial guess for the values of the parameters would provide much better starting estimates and would reduce the amount of computation to carry out to obtain a good final estimate. Thus, our method is based on constructing partial paths, instead of trying to reconstruct the entire solution path.

3.5 Combining path-following with a VNS method

The partial path method introduced in Section 3.4 is used to find the best C^* among the parameters corresponding to the path along parameter direction t_1, starting from t_0. This value depends on the choice of t_0 and t_1. In this Section, we describe a VNS-based procedure [Gendreau and Potvin, 2010, Hansen and Mladenović, 2001, Hansen et al., 2010] to obtain values for t_1 yielding good values for the parameters and good solutions for the multi-class SVM problem, in a systematic manner.

We consider a good solution to be one such that, if the real misclassification costs are known, yields lower misclassification costs on the training set. If the real misclassification costs are unknown, we search for a solution with fewer classification errors on the training set.

To complete the search procedure, we need to obtain:

- An initial direction t_{10}: If we know the real misclassification costs $\kappa^{pq}, q \neq p, p, q \in G$, we can start the partial path with $t_0^{pq} = \kappa^{pq}/\kappa$ and $t_{10}^{pq} = t_0^{pq}$, where $\kappa = \sum_{q \neq p, p, q \in G} \kappa^{pq}$.

  If we don't know the real misclassification costs, we suggest taking the initial direction $t_0 = \vec{1}$ and $t_{10} = 0.01 \times \vec{1}$, as it is equivalent to the value used in general multi-class support vector machines [Weston and Watkins, 1998] with the tradeoff parameter set as $1 + 0.01C$, although it does not consider the different misclassification costs. This is equivalent to assuming all the misclassification costs $\kappa^{pq} = 1, q \neq p, p, q \in G$.

  Starting from the initial point t_0 along t_{10}, we generate the corresponding partial path. To denote all the joints of the partial path, we introduce $PP(t_0, t_{10}) = \{C_k(t_0, t_{10}),\ k \geq 0\}$, where $C_k(t_0, t_{10})$ is the $k-$th joint. From the solutions obtained at the joints of $PP(t_0, t_{10})$, we choose the value C_{0*} corresponding to the best misclassification costs on training set I. Let

$$PI\kappa(C_{0*}, t_{10}) = \sum_{i \in I} \sum_{q \neq p, p, q \in G} \kappa^{pq} \#\{y_i = p, \hat{y}_i = q\}, \tag{3.16}$$

  to denote the lowest misclassification costs on the training set along the partial path $PP(t_0, t_{10})$, where $\hat{y}_i$ is the class membership of x_i determined by the corresponding trained support vector machine. Note that when $\kappa^{pq} = 1$, problem (3.16) returns the classification error.

- A neighborhood structure: $N_\iota(t_{10}) = \{t_1 \big| \left| \frac{t_1^{pq}}{t_{10}^{pq}} - 1 \right| \leq 0.01\iota\}$ and $\iota \leq \iota_{max}$. Although most researchers use $\iota \leq 2$ Hansen and Mladenović [2001], in the experiments of this paper, we have chosen $\iota_{max} = 10$. This choice has been made to consider a larger number of alternatives given its reduced computational cost, compared to that of other proposals.

- The algorithm will stop if at least one of these conditions is met:

  - SC1: the number of local searches ls reaches the maximum ls^{max}. In our experiments we set $ls^{max} = 100$.
  - SC2: perfect classification, i.e $PI\kappa(C_{0*}, t_{10}) = 0$.

Then we proceed the follows:

- Let $ls = 0$.
- While $ls \leq 100$ and $PI\kappa(C_{0*}, t_{10}) > 0$, repeat the following steps:
  - Step 1: Set $\iota = 1$;
  - Step 2: Until $\iota = \iota_{max}$, repeat the following steps:

- Step 2.a: Generate a direction $\tilde{t}_{10}$ at random from the ι-th neighborhood of t_{10};
- Step 2.b: Conduct a local search using $\tilde{t}_{10}$ as the initial solution by doing the following:
  - Step 2.b.1: Randomly generate $\tilde{t}_{1i}, i = 1, \cdots, N$ (we use $N = 10$ in our experiments) from the neighborhood $N_1(\tilde{t}_{10})$;
  - Step 2.b.2: For $\tilde{t}_{1i}$, generate the corresponding partial path $PP(t0, \tilde{t}_{1i})$. Along $PP(t0, \tilde{t}_{1i})$, we find

$$C^*_{1i} = \arg \min_{C \in PP(t0, \tilde{t}_{1i})} \{PI\kappa(C, \tilde{t}_{1i})\};$$

  - Step 2.b.3: Select

$$(C_{1*}, \tilde{t}_{1*}) = \arg \min_{(C^*_{1i}, \tilde{t}_{1i}), i=1,\cdots,10} \{PI\kappa(C^*_{1i}, \tilde{t}_{1i})\};$$

  - Step 2.b.4: Let $ls = ls + 1$.
- Step 2.c: If $PI\kappa(C_{1*}, \tilde{t}_{1*}) < PI\kappa(C_{0*}, t_{10})$, take $t_{10} = \tilde{t}_{1*}, C_{0*} = C_{1*}$ and go back to Step 2.b, otherwise set $\iota = \iota + 1$ and go back to Step 2.a.

3.6 Numerical experiments

To test the efficiency and reliability of this parametric partial path algorithm, we have conducted several computational experiments. All these experiments have been implemented on a Macbook with 8 gigabytes of memory, using code written in R. The RMOSEK package has been used to solve the quadratic problem (3.1) defining the start point of the algorithm.

These experiments have been conducted on the following benchmark datasets: IRIS, WINE, SEEDS, VEHICLE, CAR (Car Evaluation), GLASS, SCC (Synthetic Control Chart Time Series) and CTG (Cardiotocography, raw data). All of them are available in the UCI Machine Learning Repository. A summary of the information for these data sets is listed in Table 2.1.

Table 3.1 presents the classification performance obtained from the solutions computed at the starting point. As the real costs are unknown, we start with $t_0 = \vec{1}$ and $C = 0$. This means that at the starting point we don't consider the differences among misclassification costs, maximizing the pairwise functional margins and minimizing the training classification errors using the same weights for all of them.

From Table 3.1, we can see that for data sets WINE, SCC, CTG and these starting values, we already find the best parameters, with the best training classification accuracies. So for these data sets we don't need to construct partial paths to find better parameters.

For each of the remaining data sets, we construct a partial path starting with $C = 0$ and $t_1 = 0.01 \times \vec{1}$, as we don't know the real misclassification costs; we assume that these misclassification

Table 3.1: Classification accuracies at the starting point

Data set	IRIS	WINE	SEEDS	VEHICLE	CAR	GLASS	SCC	CTG
tr.ac	0.9831	**1**	0.9464	0.8711	0.9226	0.6488	**1**	**1**
te.ac	0.9667	0.9730	0.9048	0.8129	0.7457	0.6444	0.9833	0.9814

[1] tr.ac= classification accuracy on training set,
[2] te.ac= classification accuracy on testing set.

costs are same. We also tried some randomly chosen values for t_0 and t_1. The corresponding results show the reliability of the partial path algorithm; for details, see Appendix B.

To test the reliability of the partial paths, at each joint we compute the corresponding solutions of problem (3.3) using RMOSEK. The following figures (from Figure 3.1 to Figure 3.8) compare the solutions obtained from the partial path algorithm and directly from the optimization code. We should mention that, for both the CAR and VEHICLE data sets, the coefficient matrices are singular and do not satisfy condition $\mathcal{C}1$, as defined in Section 3.3. For these data sets we have projected the original data onto a higher-dimension space in which we obtain nonsingular coefficient matrices. We have used a modification of a nearest neighbor rule [Cover and Hart, 1967] to modify the data and to avoid this problem, in the following way:

- For the CAR data set, we add the euclidean distances from the instances to each of the means of I_1, I_2, I_3 and I_4, as additional coordinates for each instance;
- For the VEHICLE data set, we add the euclidean distances from the instances to the means, 25% quantiles and 75% quantiles of I_1, I_2, I_3 and I_4 as additional coordinates.

From Figure 3.1 to Figure 3.8, we can see that the **PPPA** procedure is very reliable, as the solutions at the joints computed from the optimization problem and **PPPA** are quite similar. Notice that using **PPPA** we need to solve only one quadratic problem (3.1), while optimization methods repeatedly solve problem (3.3) for different values of the parameters. Thus, **PPPA** offers significant savings in computation times. We show the execution times used for constructing the partial paths and completing the optimizations at all the joints of the paths in Table 3.2.

Table 3.2: Time used for getting solutions at the joints

Data set	IRIS	SEEDS	VEHICLE	CAR	GLASS
t.op(s)	5.102	12.801	2016.6	1438.690	104.864
t.pp(s)	**0.207**	**0.338**	**92.418**	**16.545**	**9.480**

[1] t.op= time used for completing the optimizations at all the joints,
[2] t.pp= time used for constructing the partial path.

Note additionally that a traditional grid search method may miss some good solutions, and will offer in general solutions of lower quality than **PPPA**.

Figure 3.1: Compare the solutions gotten from optimization and partial path algorithm for IRIS data

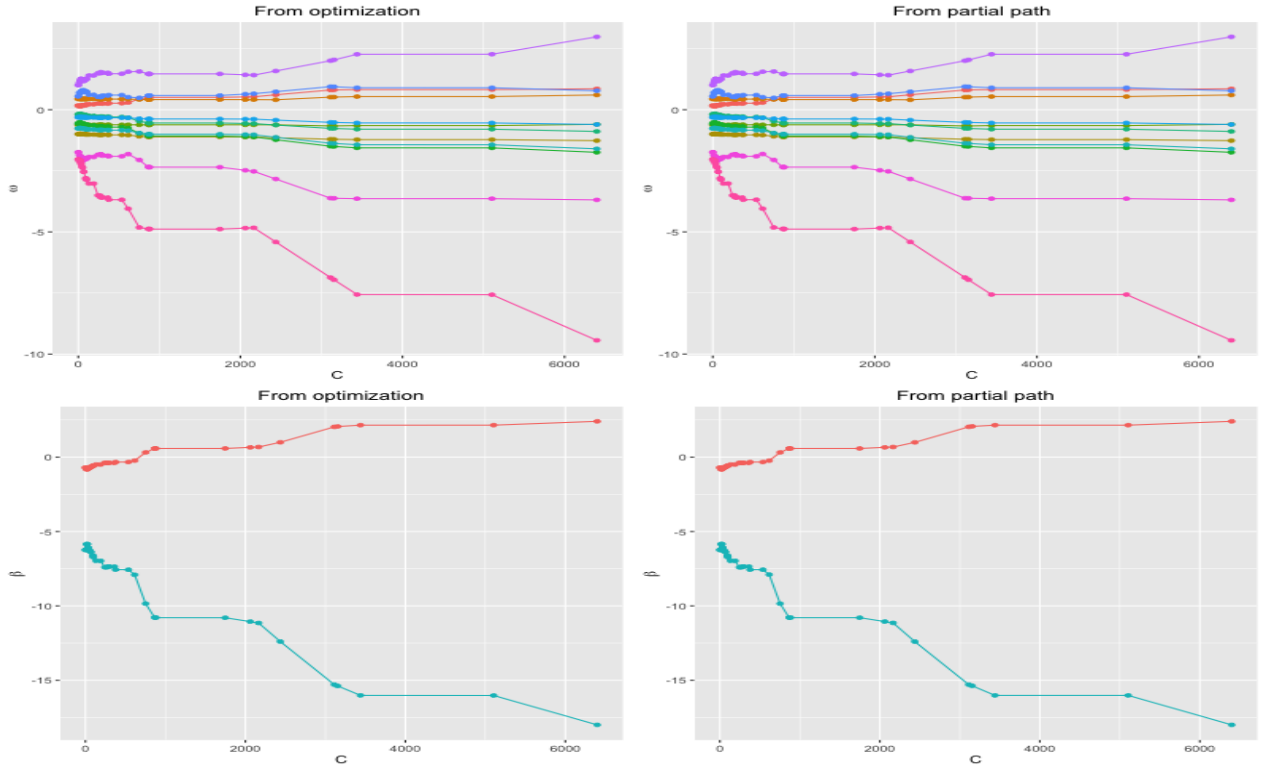


Figure 3.2: Compare the solutions gotten from optimization and partial path algorithm for SEEDS data

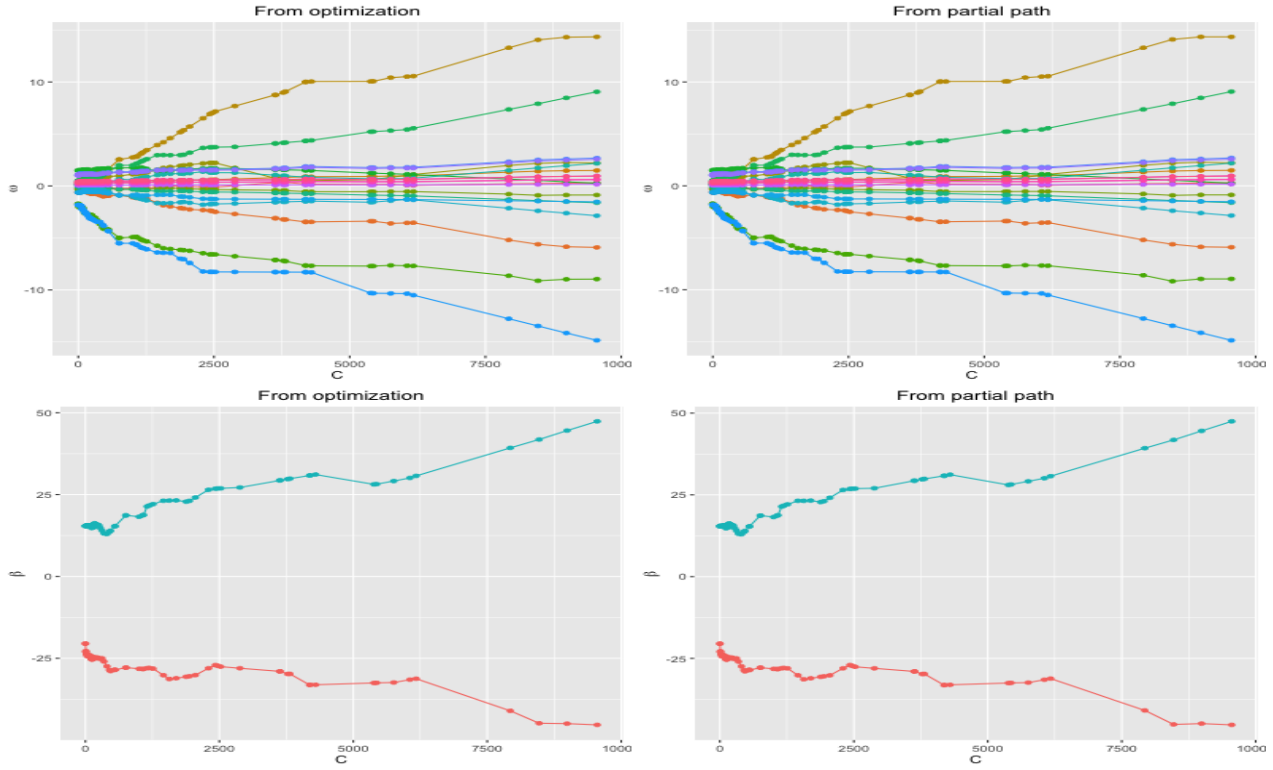


Figure 3.3: Compare the solutions gotten from optimization and partial path algorithm for CAR data

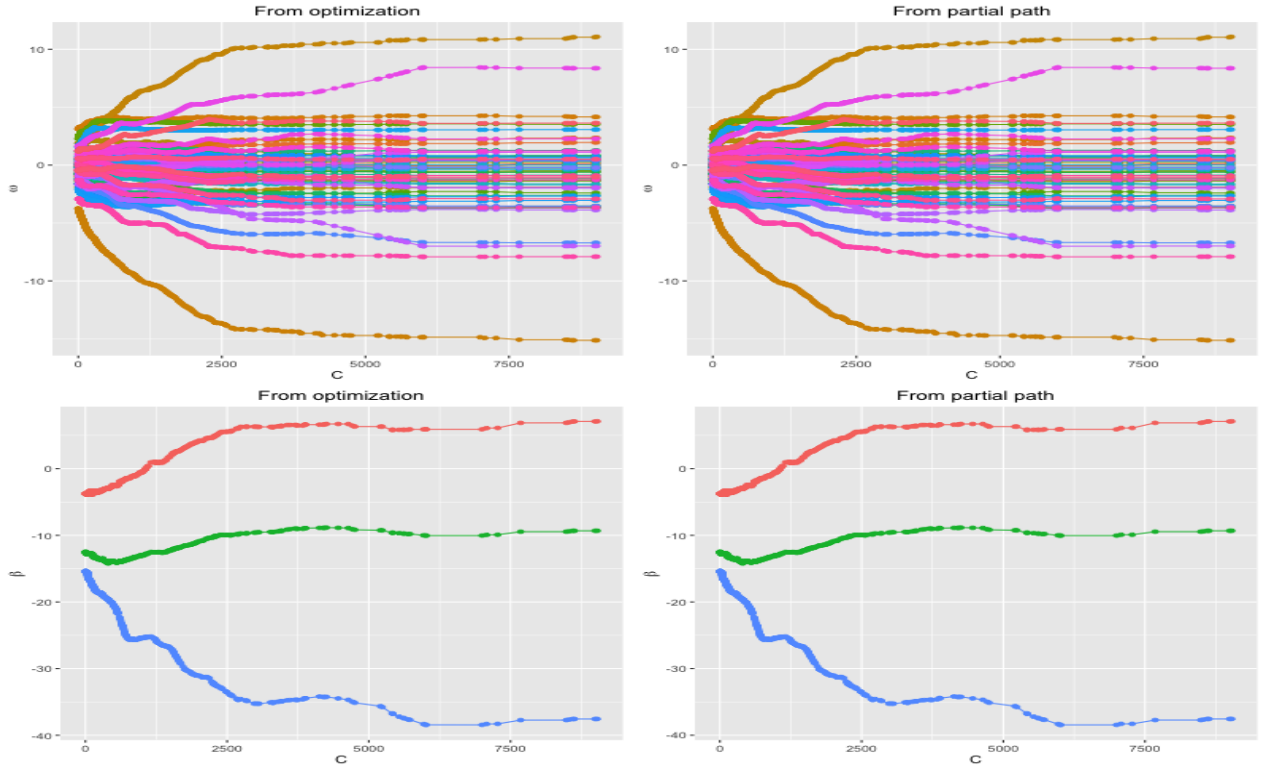


Figure 3.4: Compare the solutions gotten from optimization and partial path algorithm for VEHICLE data

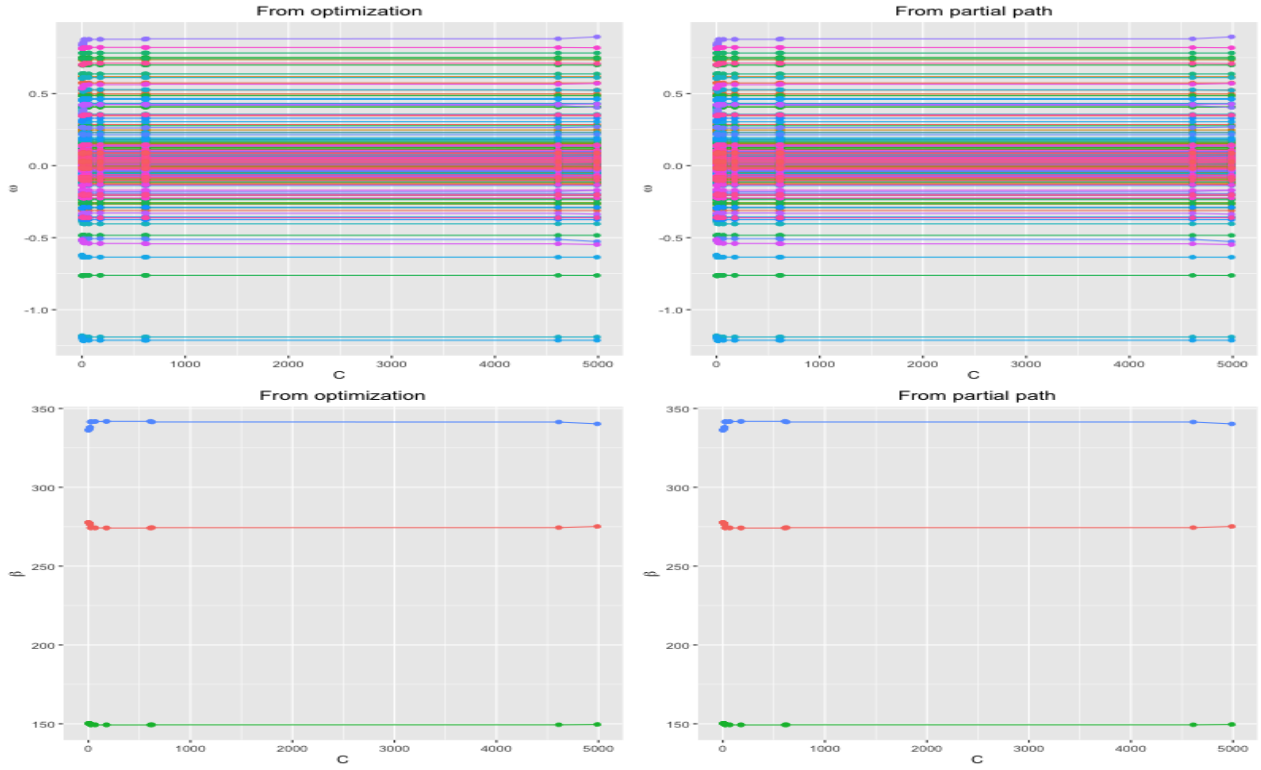


Figure 3.5: Part1: Compare the solutions gotten from optimization and partial path algorithm for GLASS data

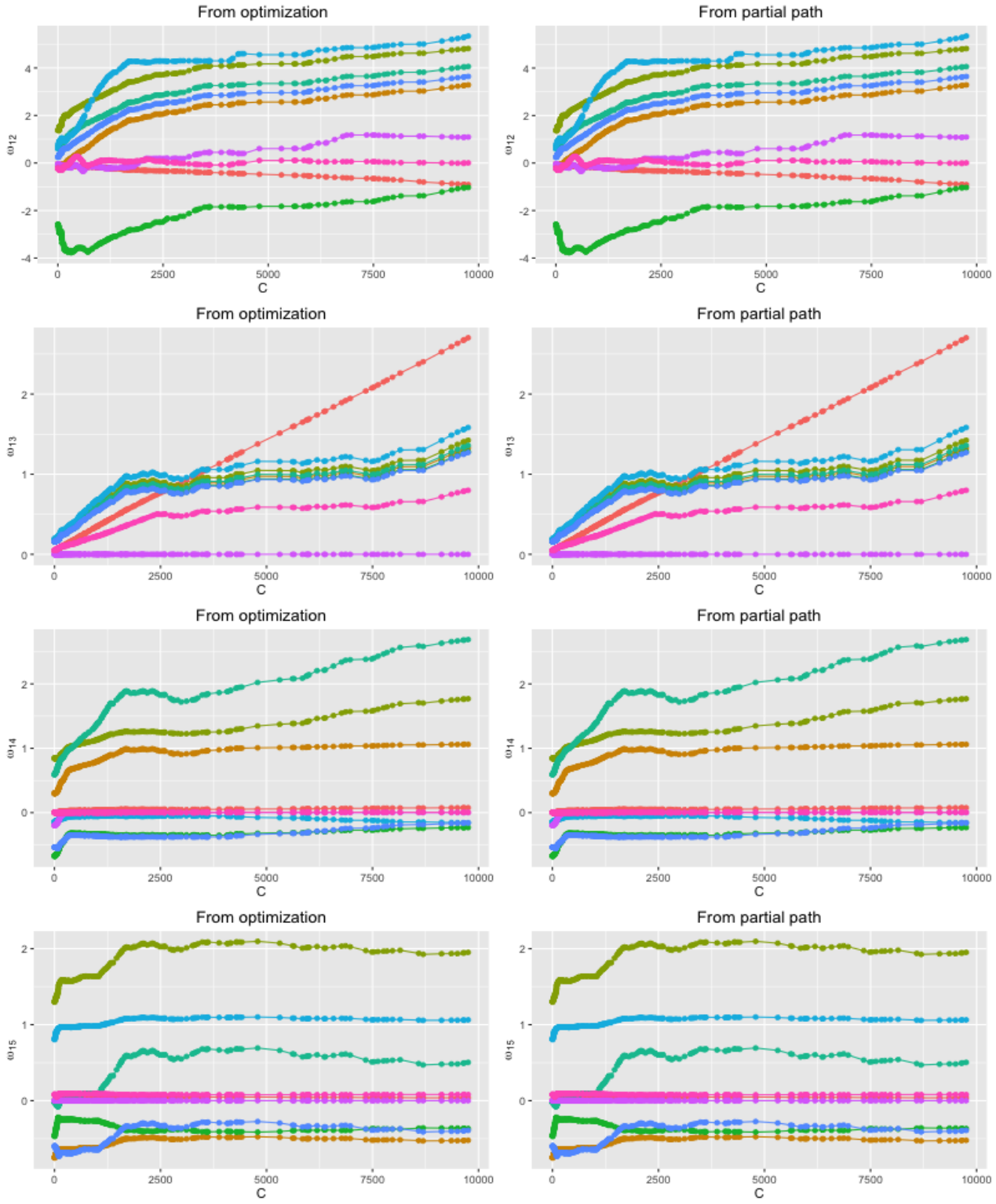


Figure 3.6: Part2: Compare the solutions gotten from optimization and partial path algorithm for GLASS data

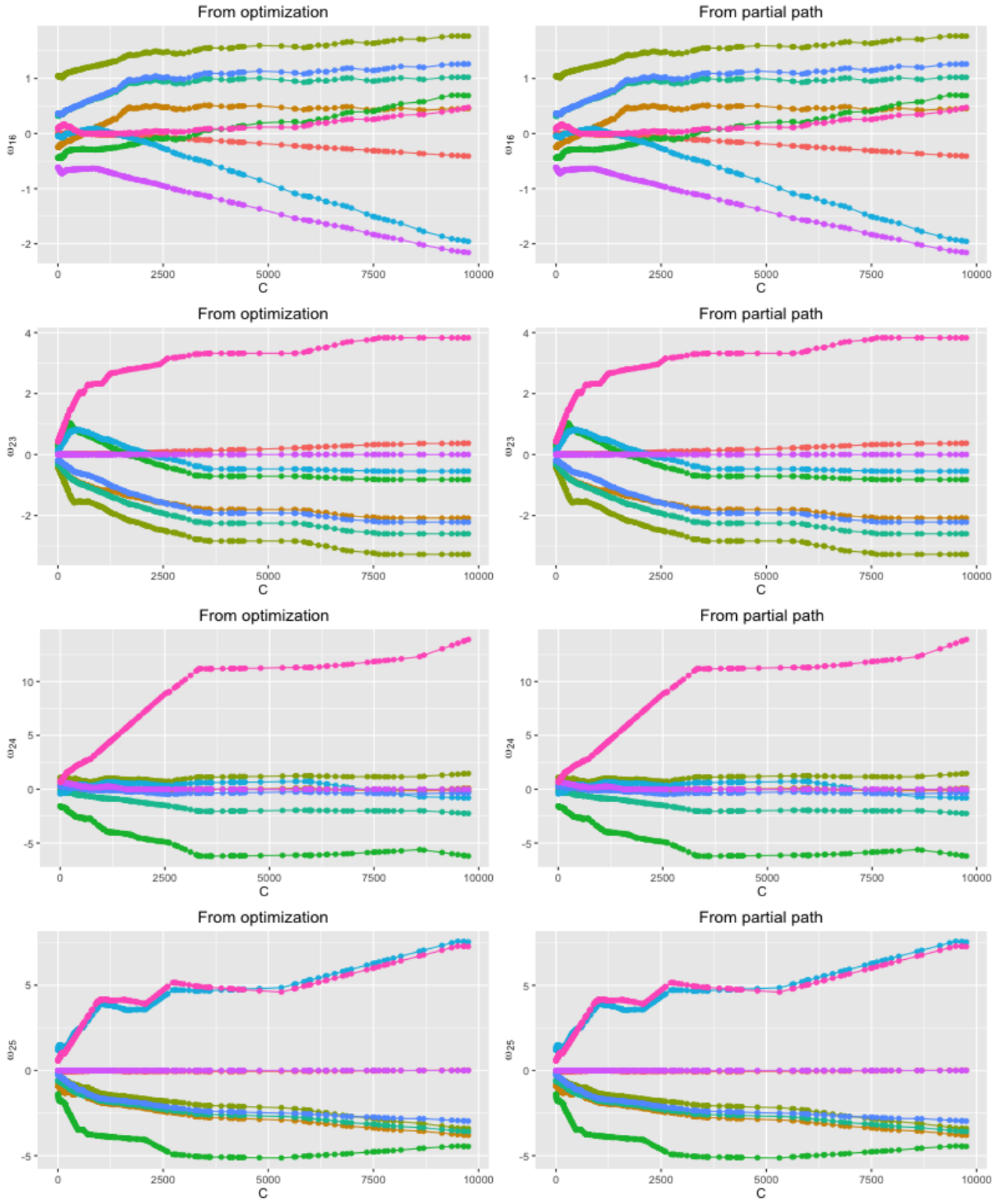


Figure 3.7: Part3: Compare the solutions gotten from optimization and partial path algorithm for GLASS data

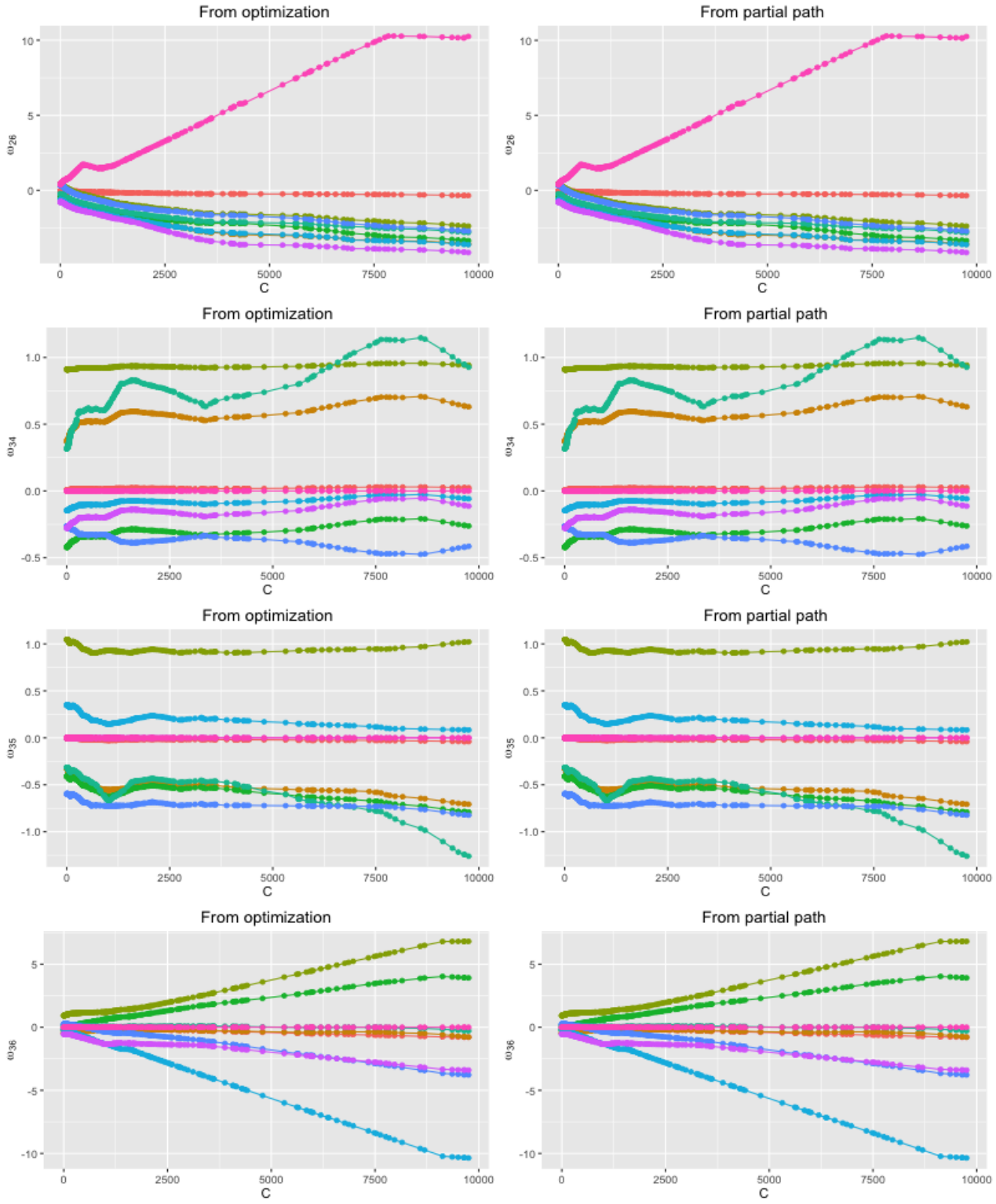
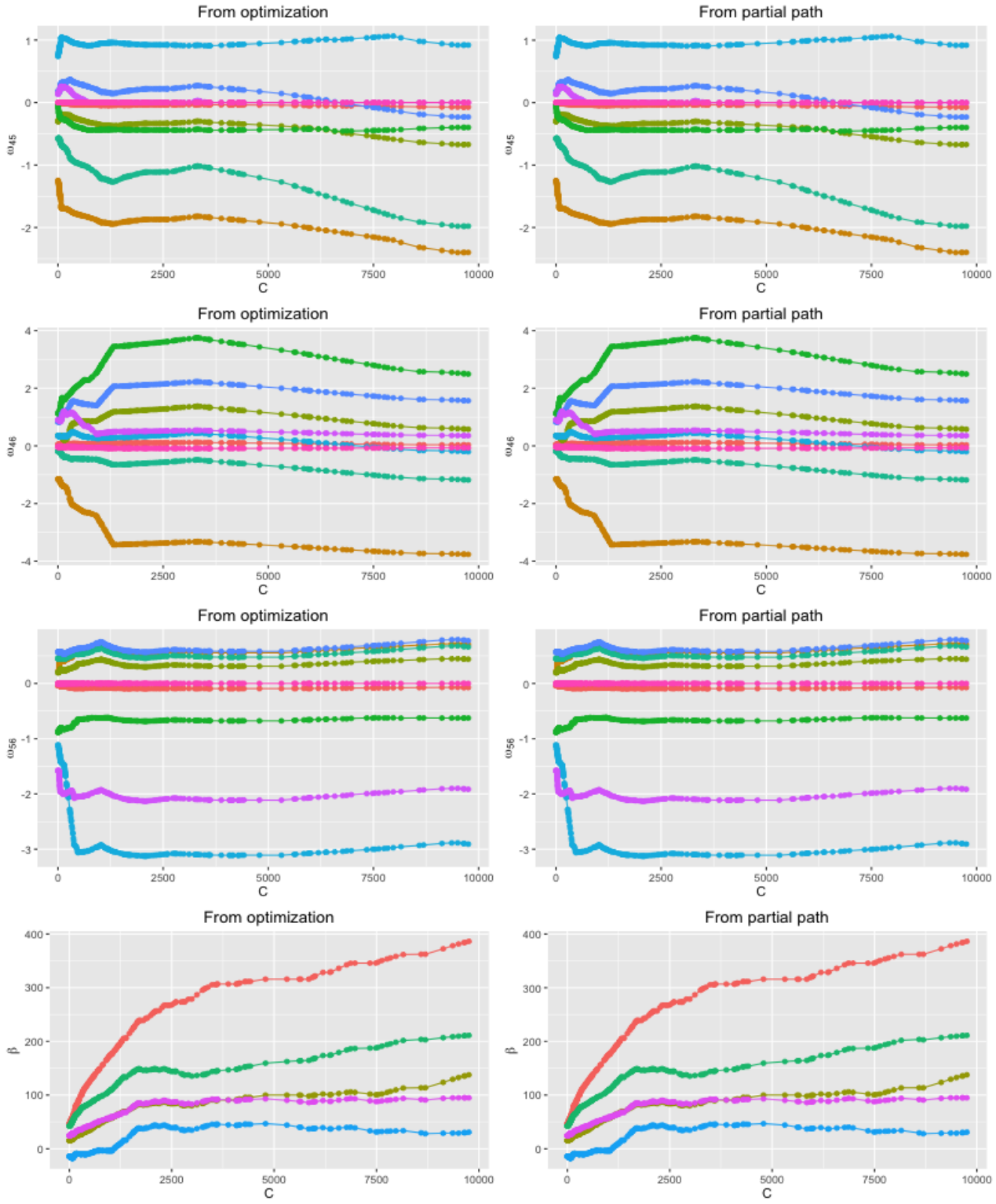


Figure 3.8: Part4: Compare the solutions gotten from optimization and partial path algorithm for GLASS data



Up to this point, we have constructed paths using $t_1 = 0.01 \times \vec{1}$. We now introduce a variable neighborhood search to find better values of the parameters. The results are shown in Table 3.3.

Table 3.3: Results obtained before and after performing a variable neighborhood search

		Unknown misclassification costs			With simulated misclassification costs		
		b.C	ac.tr	ac.te	b.C	cc.tr	cc.te
IRIS	before	22.8497	0.9915	0.9667	22.8497	6	6
	after	22.8497	0.9915	0.9667	22.8497	6	6
SEEDS	before	7930.496	0.9940	0.9286	7930.496	2	7
	after	7930.496	0.9940	0.9286	7930.496	2	7
VEHICLE	before	22.9161	0.8740	0.8129	22.9161	135	65
	after	24.2008	**0.8756**	0.8129	25.0835	**133**	65
CAR	before	7678.431	0.9378	0.8266	7678.431	548	356
	after	7678.431	0.9378	0.8266	2590.884	**452**	329
GLASS	before	4560.535	0.8274	0.62222	4560.535	209	109
	after	6805.878	**0.8333**	0.6222	9935.552	**201**	118

[1] b.C= Best parameter C found among the paths,
[2] cc.tr= classification cost on the training set,
[3] cc.te= classification cost on the testing set.

From Table 3.3, we can see that for asymmetric training data (VEHICLE, GLASS) the variable neighborhood search provides better solutions with higher classification accuracies.

We have also conducted some experiments to study the case when classification costs are known and different. We simulated values for the real misclassification costs as integers randomly generated in the interval $(1, 10)$. We compare the performance of the proposed method before and after the variable neighborhood search in Table 3.3. We can see that when there are different misclassification costs, the proposed variable neighborhood search method provides better parameter values. This would imply that when misclassification costs are known and different, to obtain better results we should assign different weights to different misclassification errors.

3.7 Conclusion

In this chapter we have proposed a method, **PPPA**, which starting from an optimal solution given by RMOSEK for multi-class SVMs, is able to identify good values for the weights of the misclassification costs with limited computational cost.

In general, we apply the proposed partial path algorithm in the following manner:

- Obtain a starting set of values for the parameters. Based on the optimization criterion (misclassification costs, for example), decide whether we need to construct a partial path.

- Along a selected parameter direction, construct a partial path from the starting point.
- Along the preceding path, find the best parameter C value. If the classification errors are not acceptable, combine the **PPPA** and a VNS method to systematically search for better parameter values.

Compared with a traditional grid search method, the partial path algorithm only needs to solve one quadratic program (3.1), while the grid search method repeatedly solves quadratic programs for each one of the parameter values tested. Thus, **PPPA** is much more efficient. Additionally, the quality of the solutions obtained from **PPPA** is higher than those from a traditional grid search method, as the number of potential solutions considered is much higher.

The partial path is constructed by following changes in the active sets. In our experiments, we have shown that the **PPPA** is efficient and reliable, because it gives us solutions which are almost the same as the ones obtained directly from the optimization problem, while requiring at most one tenth of the computational effort. From our experiments, we also see that if the misclassification costs are different (or the training data is asymmetric), a VNS method provides significantly better parameter values.

In summary, we conclude that the **PPPA** is an efficient and reliable procedure to find good parameter values for multi-class SVMs. Combining it with a VNS method helps us to systematically and efficiently explore a very large set of potential parameter weight values.

Chapter 4

Conclusions and Future Research

4.1 Conclusion

Binary SVMs have been widely used for classification because of their efficiency and high classification accuracy. These efficient methods have been extended to multi-class classification problems; these extensions can be roughly classified into two groups: single-objective approaches and multi-objective ones. One of the main challenges associated to multi-class classification procedures, due to the combinatorial aspects of this problem, is how to construct good classifiers efficiently. For a multi-class classification problem, we need to construct at least m classifiers (exactly m classifiers when using all-together or one-against-all based SVM approaches and $m(m-1)/2$ classifiers when using a one-against-one based SVM approach). In addition to this, practical considerations require that the methods should be able to incorporate the treatment of different misclassification costs for different classes, as well as the possible existence of skewed class distributions. The adaptation of the traditional approaches to incorporate all of the preceding aspects, while obtaining good classifiers, has implied huge computational costs, especially when we have a large number of classes to consider at the same time.

In this thesis we have extended the efficient binary SVM approach to solve multi-class classification problems in two different ways: i) using a multi-objective approach (**PM**) and ii) using a weighted single-objective approach (**PPPA**). For both of these approaches, we have considered models that take into account possible differences among the classification costs. These extensions aim to provide a more general, but still computationally efficient, multi-class SVM class of models.

Specifically, for **PM** we solve a model that considers all objectives to obtain its Pareto-optimal solutions. Note that in general it is not possible to maximize the geometric margins and minimize the misclassification penalties simultaneously for any set of values of the variables. Thus, our aim is not to compute optimal solutions for **PM**. Rather, the advantage of using our model of interest, **PM**, is that by working with proper projections of the data and the quadratic penalties as introduced in Chapter 2, we have the misclassification penalties embedded with the

geometric margins (sharing the same denominators in the objective functions) and thus we can theoretically characterize its Pareto-optimal solutions based on the optimal solution of a single QP.

These Pareto-optimal solutions can be obtained as affine transformations of the initial solution with given coefficients, as presented in Corollary 2.5. Based on the simplicity of this structure, it is possible to provide decision makers with a large amount of Pareto-optimal solutions, or to select among them the best ones according to some additional criteria, with limited computational effort. We have compared numerically **PM** with other multi-class SVM approaches such as the all-together method (AT), one-against-all method (OAA), one-against-one method (OAO), multi-objective SVM based on all-together method (MS2), multi-objective SVM based on one-against-all method (SM-OA) and multi-objective SVM based on one-against-one method (SM-OAO). Our computational experiments show that **PM** is both efficient and effective on a number of test problems. Compared with single-objective approaches (AT, OAA and OAO), **PM** offers comparable or better classification accuracies with low computational cost. Compared with other multi-objective approaches (MS2, SM-OA and SM-OAO), it offers better classification accuracies and gives better approximations of the Pareto frontiers with computational times that are smaller by a factor of 60 or more.

Our second proposed model **PPPA** differs from the first procedure in that, instead of treating the problem as a multi-objective one, we proceed by constructing a weighted single-objective SVM. The main aim has been to find proper values for the weights of the penalties, in an efficient manner. For **PPPA**, we define the weights as a linear combination $(t_0 + Ct_1)$ of some initial weights t_0 and a set of search weights t_1. For given t_0 and t_1 and using linear penalties for the misclassification costs, it is possible to show that (see Theorem 3.2) when the row vectors of (X_O, R_O) are linearly independent, the optimal solution of the weighted single-objective SVM is piecewise-affine with respect to parameter C. This value defines how far we should move along t_1, from t_0.

In general, we stop the procedure with the optimal solution corresponding to the initial set of weights if a satisfying classification performance is achieved. Otherwise, we construct a partial path along some chosen direction t_1. Among the optimal solutions along the path, we find a value C yielding the best classification performance. If this classification performance attains a prespecified satisfactory level, we finish the procedure. Otherwise, we combine a VNS method with **PPPA** to systematically search for better values of the weight parameters $t_1^{pq}, q \neq p, p, q \in G$. Notice that, for **PPPA**, we only need to solve one QP at the starting set of parameter values. The solutions at the joints defining the piecewise linear portions of the path can be computed by monitoring the active sets for different values of the parameter C, and solving some systems of linear equations. Thus, **PPPA** can be implemented with very low overall computational costs, requiring the solution of one QP problem and several systems of linear equations of reasonable size (that of the support sets). In Chapter 3, we have also numerically shown that **PPPA** is reliable and efficient, in the sense that the optimal solutions offered by **PPPA** are almost always the same as the ones obtained from directly solving the

corresponding optimization problem. It also has much lower computational costs: from our numerical experiments, **PPPA** is at least 11 times faster than a grid search method which considers all the joints along the path.

4.2 Future Research Lines

As a consequence of the results obtained in the derivation of the methods proposed in this dissertation, we have identified several possible research lines to be pursued in the near future:

1. We have noticed that the **PPPA** method as introduced requires some starting penalty values to compute an initial optimal solution. This optimal solution is obtained by solving a weighted single-objective SVM (a QP problem) which obtains all the $m(m-1)/2$ classifiers of an all-together method, at the same time. As we have already mentioned, an all-together method (a single-objective SVM considering all the classes at the same time) is limited to small problems because of its computational solution costs and memory requirements [Hsu and Lin, 2002, Weston and Watkins, 1999]. But due to the rapidly increasing data base sizes in recent years, we would wish to be able to handle very large QP problems. A sequential minimal optimization (SMO) approach has been introduced in [Platt, 1999] to solve large-scale binary SVM problems. Numerical experiments in [Platt, 1999] show the efficiency of SMO when solving problems with up to tens of thousands of training objects. We would like to extend this effective SMO approach to solve large-scale single-objective multi-class SVM problems such as those of interest for the proposed methods.

2. In this thesis, we have used different forms of the penalties terms in the objective function for our two proposed models. Compared with **PPPA**, using **PM** we obtain a large number of Pareto-optimal solutions without specifying direct weights for the penalties. For **PM**, we have used the quadratic form of the penalties in order to ensure that all the geometric margins have the same denominator. When there are redundant noise features, L_1-norm SVMs may present advantages over models based on the use of L_2-norms, [Zhu et al., 2004]. Several numerical experiments also show advantages of L_1-norm SVMs over L_2-norm SVMs from the point of view of their classification performance, [Wang and Shen, 2007]. We are interested in extending our efficient **PM** method to the use of L_1-norm versions using linear penalties.

3. We are also interested in developing adapted methods, based on the ideas presented in the thesis, for specific applications of SVM classifiers. SVM has been used successfully in many real-world problems: text categorization [Joachims, 1998], image classification [Chapelle et al., 1999], protein classification [Ding and Dubchak, 2001], cancer classification [Guyon et al., 2002], hand-written character recognition [Bahlmann et al., 2002], time series prediction [Kim, 2003, Van Gestel et al., 2001], stock selection[Fan and Palaniswami,

2001] and so on. In our experiments, we have considered only benchmark multi-class classification problems. Our goal would be to be able to adapt the proposed methods to much larger data sets from specific application areas in order to solve very large and complex multi-class classification problems, by taking advantage of their special structures and characteristics.

Appendix A

Appendix to Chapter 2

A.1 Proof of Lemma 2.1

First, assume that (ω^*, β^*) is optimal for problem (2.3). Notice that the feasible region of problem (2.3) and the feasible region of problem (hard-margin **PM**) are the same.

If (ω^*, β^*) is not weakly Pareto-optimal for problem (hard-margin **PM**), there will exist a feasible (ω_0, β_0) such that

$$\varrho^{12}(\omega_0, \beta_0) > \varrho^{12}(\omega^*, \beta^*), \varrho^{21}(\omega_0, \beta_0) > \varrho^{21}(\omega^*, \beta^*) \cdots, \varrho^{m(m-1)}(\omega_0, \beta_0) > \varrho^{m(m-1)}(\omega^*, \beta^*).$$

As $\theta^{pq} > 0$, we have:

$$\begin{aligned} &\varrho^{12}(\omega_0, \beta_0) > \varrho^{12}(\omega^*, \beta^*), \theta^{21}\varrho^{21}(\omega_0, \beta_0) > \theta^{21}\varrho^{21}(\omega^*, \beta^*), \cdots, \\ &\theta^{m(m-1)}\varrho^{m(m-1)}(\omega_0, \beta_0) > \theta^{m(m-1)}\varrho^{m(m-1)}(\omega^*, \beta^*). \end{aligned}$$

This contradicts our assumption that (ω^*, β^*) is optimal for problem (2.3).

As a consequence, (ω^*, β^*) must be a weakly Pareto-optimal solution of problem (hard-margin **PM**). Then, for any feasible (ω_0, β_0), there exists some $i \neq j, i, j \in G$ such that $\varrho^{ij}(\omega_0, \beta_0) \leq \varrho^{ij}(\omega^*, \beta^*)$. Let

$$\varrho_* = \max\left(\varrho_*^{12}, \varrho_*^{21}, \cdots, \varrho_*^{(m-1)m}, \varrho_*^{m(m-1)}\right),$$

where $\varrho_*^{pq} = \varrho^{pq}(\omega^*, \beta^*), p \neq q, p, q \in G$.

Formulate the following problem:

$$\begin{aligned} \max_{\omega,\beta} \min &\left(\frac{\varrho_*}{\varrho_*^{12}}\varrho^{12}(\omega, \beta), \frac{\varrho_*}{\varrho_*^{21}}\varrho^{21}(\omega, \beta), \cdots, \frac{\varrho_*}{\varrho_*^{m(m-1)}}\varrho^{m(m-1)}(\omega, \beta)\right), \\ \text{s.t.} \quad &(\omega^{pq})^T x_i + \beta^{pq} > 0, \quad i \in I_p, p < q, p, q \in G, \\ &-(\omega^{pq})^T x_i - \beta^{pq} > 0, \quad i \in I_q, p < q, p, q \in G. \end{aligned} \tag{A.1}$$

It is easy to see that (ω^*, β^*) is optimal for problem (A.1). By diving all the objectives in problem (A.1) by $\frac{\varrho_*}{\varrho_*^{12}}$, we get the equivalent optimization problem:

$$\begin{aligned} \max_{\omega,\beta} \min & \left(\varrho^{12}(\omega,\beta), \frac{\varrho_*^{12}}{\varrho_*^{21}}\varrho^{21}(\omega,\beta), \cdots, \frac{\varrho_*^{12}}{\varrho_*^{m(m-1)}}\varrho^{m(m-1)}(\omega,\beta) \right), \\ \text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} > 0, \quad i \in I_p, p < q, p, q \in G, \\ & -(\omega^{pq})^T x_i - \beta^{pq} > 0, \quad i \in I_q, p < q, p, q \in G. \end{aligned} \tag{A.2}$$

Thus, (ω^*, β^*) is also optimal for problem (A.2).

A.2 Proof of Theorem 2.4

As before, the weakly Pareto-optimal solution of **SPMAT** can be found by solving the following problem:

$$\begin{aligned}
\max_{\omega,\beta,\xi}\min \quad & \Big(\bar{\varrho}^{12}(\omega,\beta), \theta^{21}\bar{\varrho}^{21}(\omega,\beta), \cdots, \theta^{(m-1)m}\bar{\varrho}^{(m-1)m}(\omega,\beta), \theta^{m(m-1)}\bar{\varrho}^{m(m-1)}(\omega,\beta),\Big) \\
\text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} + \xi_i^{pq} > 0, \quad i \in I_p, q > p, p, q \in G, \\
& -(\omega^{pq})^T x_i - \beta^{pq} + \xi_i^{qp} > 0, \quad i \in I_p, q > p, p, q \in G, \\
& \xi_i^{pq} \geq 0, \quad i \in I_p, p \neq q, p, q \in G.
\end{aligned} \tag{A.3}$$

Problem problem (A.3) is equivalent to

$$\begin{aligned}
\min_{\omega,\beta,\xi} \quad & \frac{\|(\omega, \sqrt{C}\xi)\|}{\min\{\min\limits_{i\in I_1}(\omega^{12})^T x_i + \beta^{12} + \xi_i^{12}, \cdots, \theta^{m(m-1)} \min\limits_{i\in I_m}(\omega^{m(m-1)})^T x_i + \beta^{m(m-1)} + \xi_i^{m(m-1)}\}}, \\
\text{s.t.} \quad & (\omega^{pq})^T x_i + \beta^{pq} + \xi_i^{pq} > 0, \quad i \in I_p, q > p, p, q \in G, \\
& -(\omega^{pq})^T x_i - \beta^{pq} + \xi_i^{qp} > 0, \quad i \in I_p, q > p, p, q \in G, \\
& \xi_i^{pq} \geq 0, \quad i \in I_p, p \neq q, p, q \in G.
\end{aligned} \tag{A.4}$$

By introducing a condition to bound away from zero the denominator of the objective function, we obtain the equivalent problem

$$\begin{aligned}
\min_{\omega,\beta,\xi} \quad & \|(\omega, \sqrt{C}\xi)\|, \\
\text{s.t.} \quad & \theta^{pq}[(\omega^{pq})^T x_i + \beta^{pq} + \xi_i^{pq}] \geq 1, \quad i \in I_p, q > p, p, q \in G, \\
& \theta^{qp}[-(\omega^{pq})^T x_i - \beta^{pq} + \xi_i^{qp}] \geq 1, \quad i \in I_q, q > p, p, q \in G, \\
& \xi_i^{pq} \geq 0, \quad i \in I_p, p \neq q, p, q \in G.
\end{aligned} \tag{A.5}$$

Problem (A.5) is also equivalent to

$$\begin{aligned}
\min_{\omega,\beta,\xi} \quad & \|(\omega, \sqrt{C}\xi)\|^2, \\
\text{s.t.} \quad & \theta^{pq}[(\omega^{pq})^T x_i + \beta^{pq} + \xi_i^{pq}] \geq 1, \quad i \in I_p, q > p, p, q \in G, \\
& \theta^{qp}[-(\omega^{pq})^T x_i - \beta^{pq} + \xi_i^{qp}] \geq 1, \quad i \in I_q, q > p, p, q \in G, \\
& \xi_i^{pq} \geq 0, \quad i \in I_p, p \neq q, p, q \in G.
\end{aligned} \tag{A.6}$$

From the strict convexity of the objective function of problem (A.6) its optimal solution $(\omega_\theta, \xi_\theta)$ is unique. As the constraints are affine functions and the objective is quadratic (and positive definite), the KKT conditions are necessary and sufficient for optimality.

These KKT conditions are:

$$
\begin{aligned}
&2\omega_\theta^{pq} = \theta^{pq}\sum_{i\in I_p}\lambda_{\theta i}^{pq}x_i - \theta^{qp}\sum_{i\in I_q}\lambda_{\theta i}^{qp}x_i,\ \ q>p,\ \ p,q\in G,\\
&\sum_{i\in I_p}\theta^{pq}\lambda_{\theta i}^{pq} - \theta^{qp}\sum_{i\in I_q}\lambda_{\theta i}^{qp} = 0, \quad q>p,\ \ p,q\in G\\
&2c^{pq}\xi_{\theta i}^{pq} = \theta^{pq}\lambda_{\theta i}^{pq} + \tau_{\theta i}^{pq}, \quad i\in I_p,\ \ p\neq q,\ \ p,q\in G,\\
&\lambda_{\theta i}^{pq}[\theta^{pq}(\omega_\theta^{pq})^T x_i + \theta^{pq}\beta_\theta^{pq} + \theta^{pq}\xi_{\theta i}^{pq} - 1] = 0, \quad i\in I_p,\ \ q>p,\ \ p,q\in G,\\
&\lambda_{\theta i}^{qp}[-\theta^{qp}(\omega_\theta^{pq})^T x_i - \theta^{qp}\beta_\theta^{pq} + \theta^{qp}\xi_{\theta i}^{qp} - 1] = 0, \quad i\in I_q,\ \ q>p,\ \ p,q\in G,\\
&\xi_{\theta i}^{pq}\geq 0, \lambda_{\theta i}^{pq}\geq 0, \tau_{\theta i}^{pq}\geq 0, \quad i\in I_p, p\neq q, p,q\in G,\\
&\theta^{pq}[(\omega_\theta^{pq})^T x_i + \beta_\theta^{pq} + \xi_{\theta i}^{pq}]\geq 1, \quad i\in I_p, q>p, p,q\in G,\\
&\theta^{qp}[-(\omega_\theta^{pq})^T x_i - \beta_\theta^{pq} + \xi_{\theta i}^{qp}]\geq 1, \quad i\in I_q, q>p, p,q\in G,\\
&\tau_{\theta x}^{pq}[-\xi_{\theta i}^{pq}] = 0, \quad i\in I_p, q\neq p, p,q\in G.
\end{aligned}
\tag{A.7}
$$

From these conditions we can see that $(\lambda_\theta^{pq},\lambda_\theta^{qp})\neq 0, q>p,\ \ p,q\in G$. Then, there exists some $x_\theta^{pq}\in I_p$(without loss of generality), such that $\lambda_{\theta x}^{pq}>0$. So we have

$$
\beta_\theta^{pq} = \frac{1}{\theta^{pq}} - (\omega_\theta^{pq})^T x_\theta^{pq} - \xi^{pq}(x_\theta^{pq}), \quad q>p, p,q\in G.
$$

From this characterization, the set of optimal solutions for problem (A.6) is nonempty. From the convexity of the objective function, we have that problem (A.6) has a unique optimal solution. When $\theta=(1,1,\cdots,1,1)$, we have problem (A.6) $\Longleftrightarrow$ (P2).

Suppose (ω_1,β_1) is optimal for (*P2*) and (λ_1,τ_1) are the corresponding KKT multiplier vector. Then let

$$
\begin{aligned}
\omega_\theta^{pq} &= \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\omega_1^{pq}, \quad q>p,\ \ p,q\in G\\
\beta_\theta^{pq} &= \frac{\theta^{qp}-\theta^{pq}}{2\theta^{pq}\theta^{qp}} + \frac{\theta^{qp}+\theta^{pq}}{2\theta^{pq}\theta^{qp}}\beta_1^{pq}, \quad q>p,q,p\in G,\\
\xi_{\theta i}^{pq} &= \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\xi_{1i}^{pq}, \quad i\in I_p, p\neq q,\ \ p,q\in G,\\
\lambda_{\theta i}^{pq} &= \frac{\theta^{pq}+\theta^{qp}}{2\theta^{pq}\theta^{qp}}\frac{1}{\theta^{pq}}\lambda_{1i}^{pq}, \quad i\in I_p,\ \ p\neq q, p,q\in G,\\
\tau_{\theta i}^{pq} &= \frac{\theta^{pq}+\theta^{qp}}{2\theta^{qp}\theta^{pq}}\tau_{1i}^{pq}, \quad i\in I_p, p\neq q, p,q\in G.
\end{aligned}
\tag{A.8}
$$

These values $(\omega_\theta,\beta_\theta,\xi_\theta)$ are the unique optimal solution of problem (A.6), since they satisfy the KKT conditions problem (A.7).

Appendix B

Appendix to Chapter 3

B.1 Compare The Solutions Gotten from Optimization and Partial Path Algorithm, t_0, t_1 randomly chosen

Figure B.1: Compare the solutions gotten from optimization and partial path algorithm for IRIS data with randomly chosen t_0, t_1

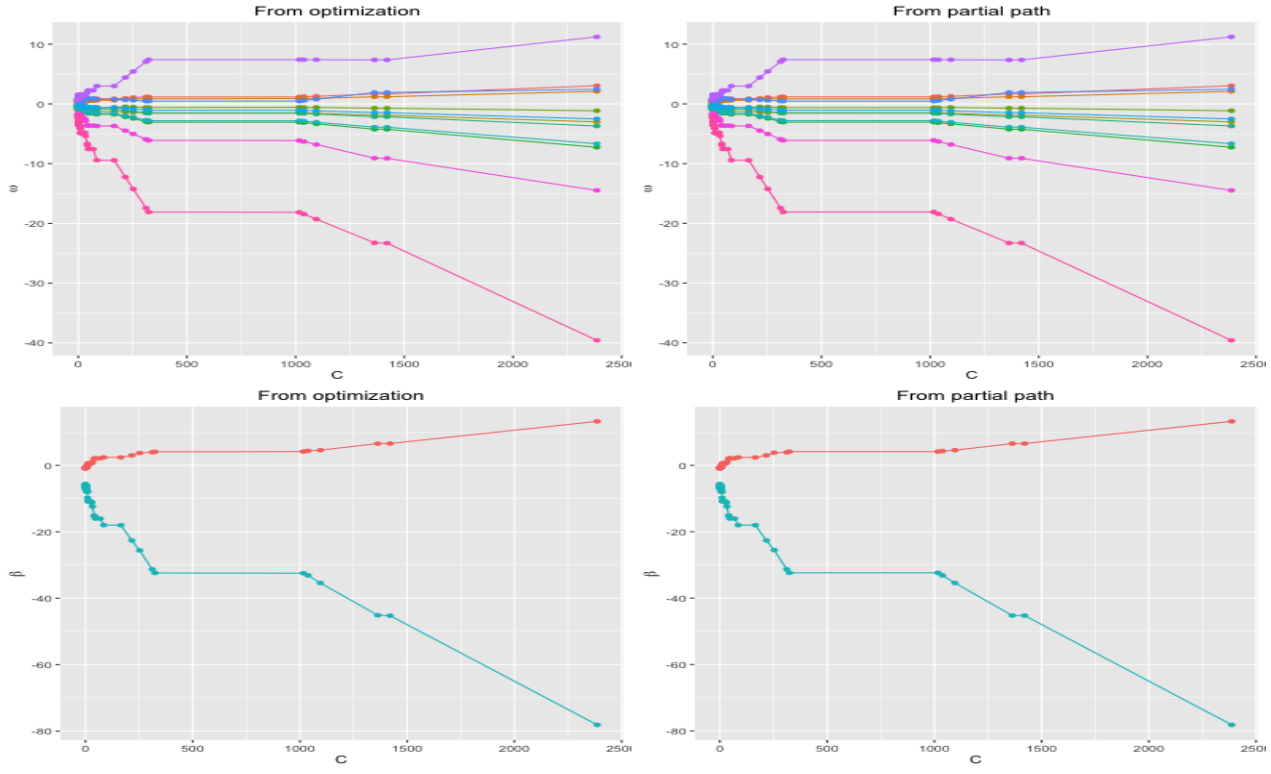


Figure B.2: Compare the solutions gotten from optimization and partial path algorithm for SEEDS data with randomly chosen t_0, t_1

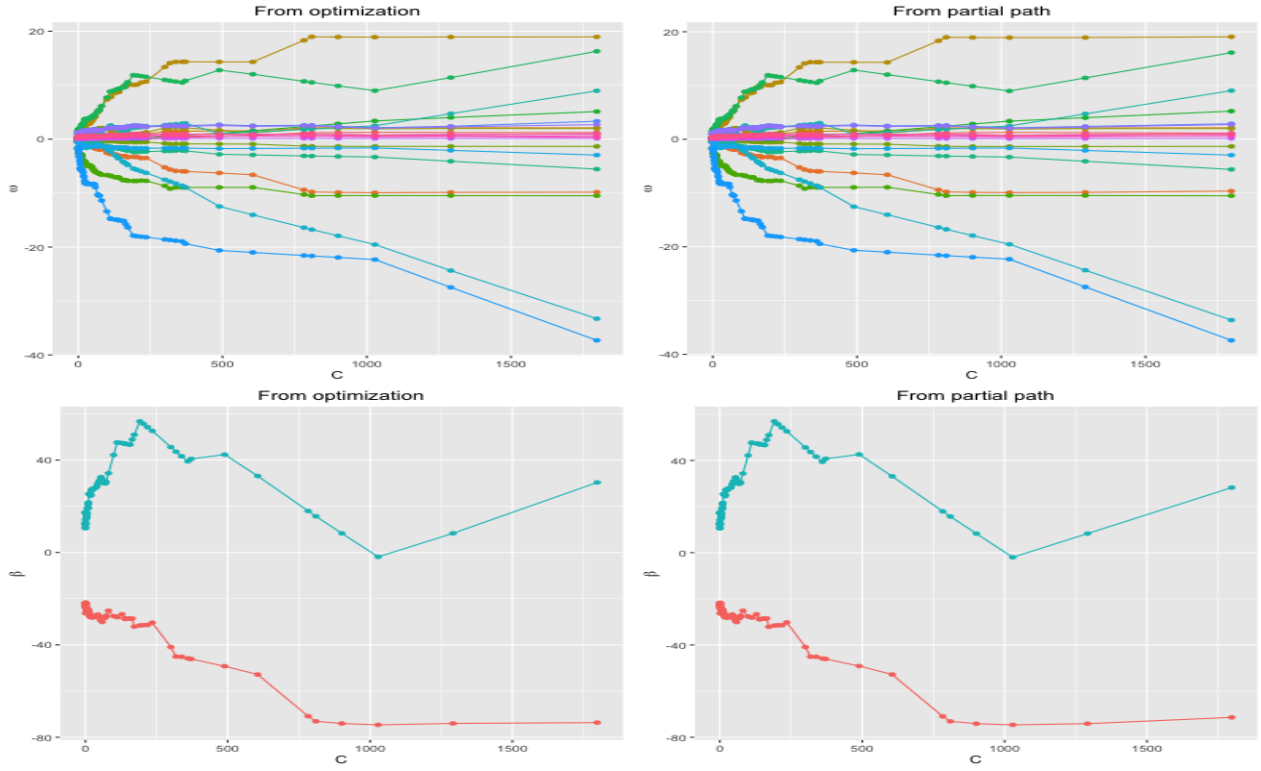


Figure B.3: Compare the solutions gotten from optimization and partial path algorithm for CAR data with randomly chosen t_0, t_1

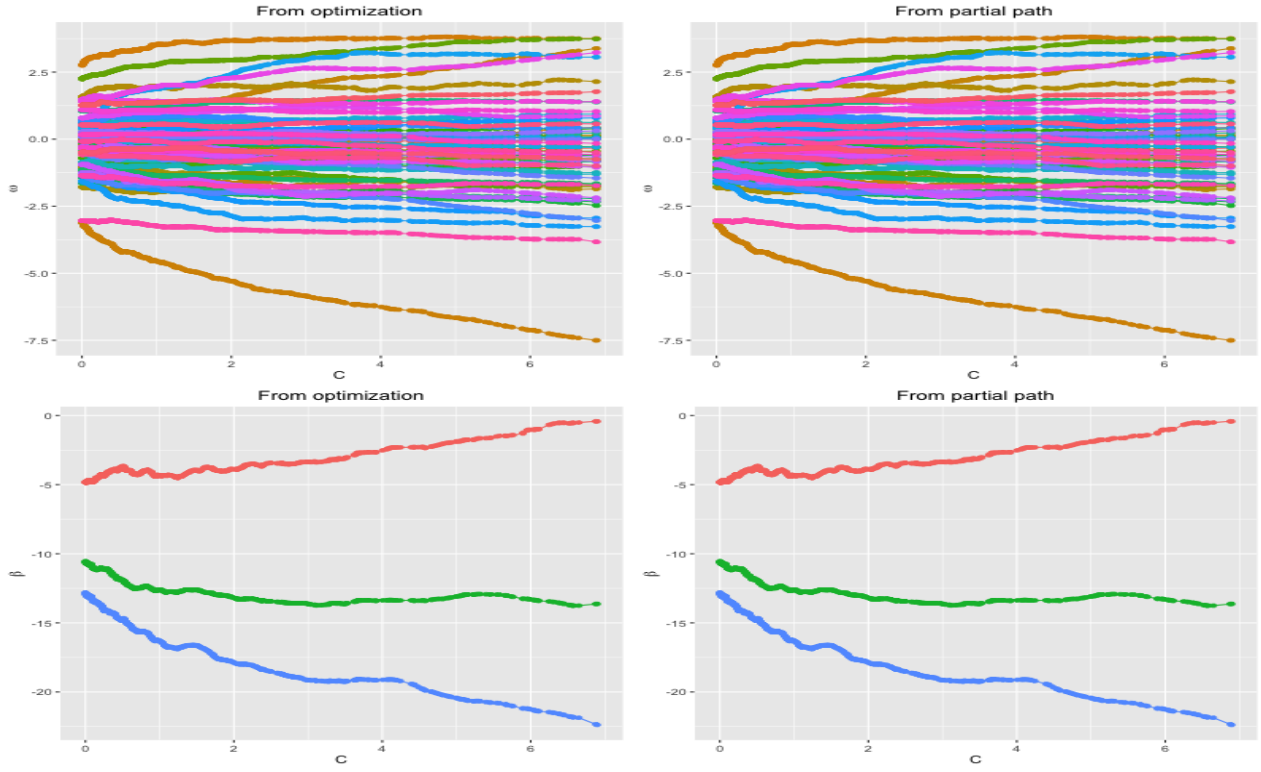


Figure B.4: Compare the solutions gotten from optimization and partial path algorithm for VEHICLE data with randomly chosen t_0, t_1

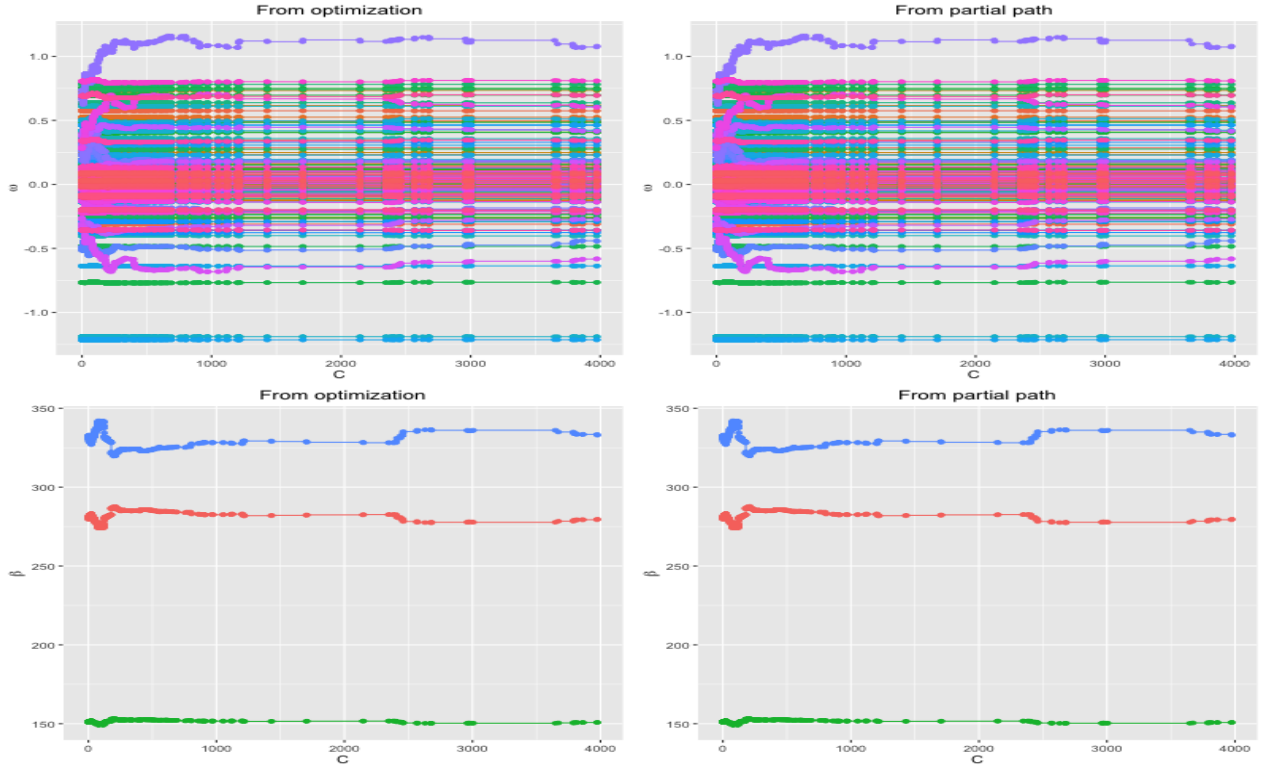


Figure B.5: Part1: Compare the solutions gotten from optimization and partial path algorithm for GLASS data with randomly chosen t_0, t_1

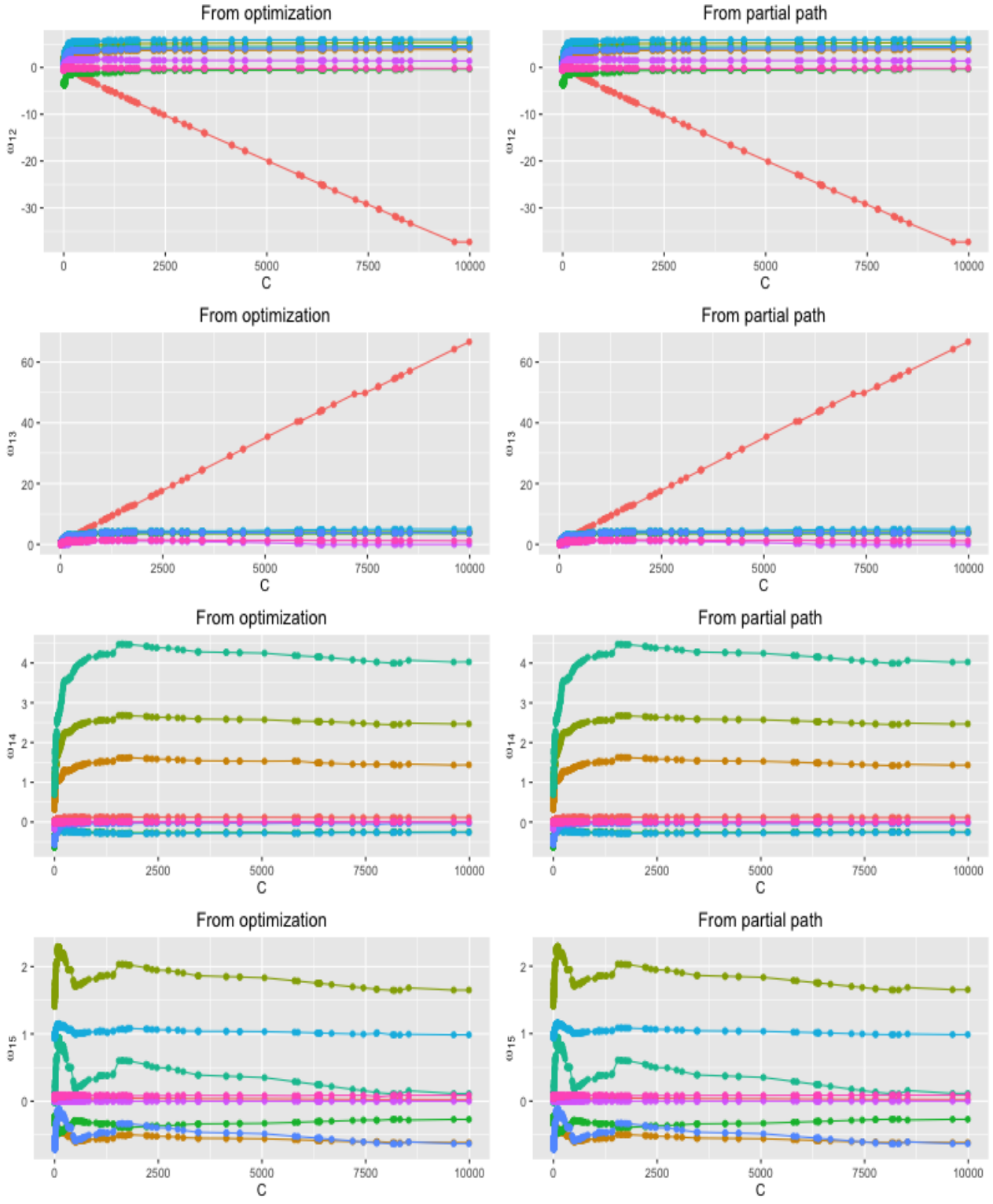


Figure B.6: Part2: Compare the solutions gotten from optimization and partial path algorithm for GLASS data with randomly chosen t_0, t_1

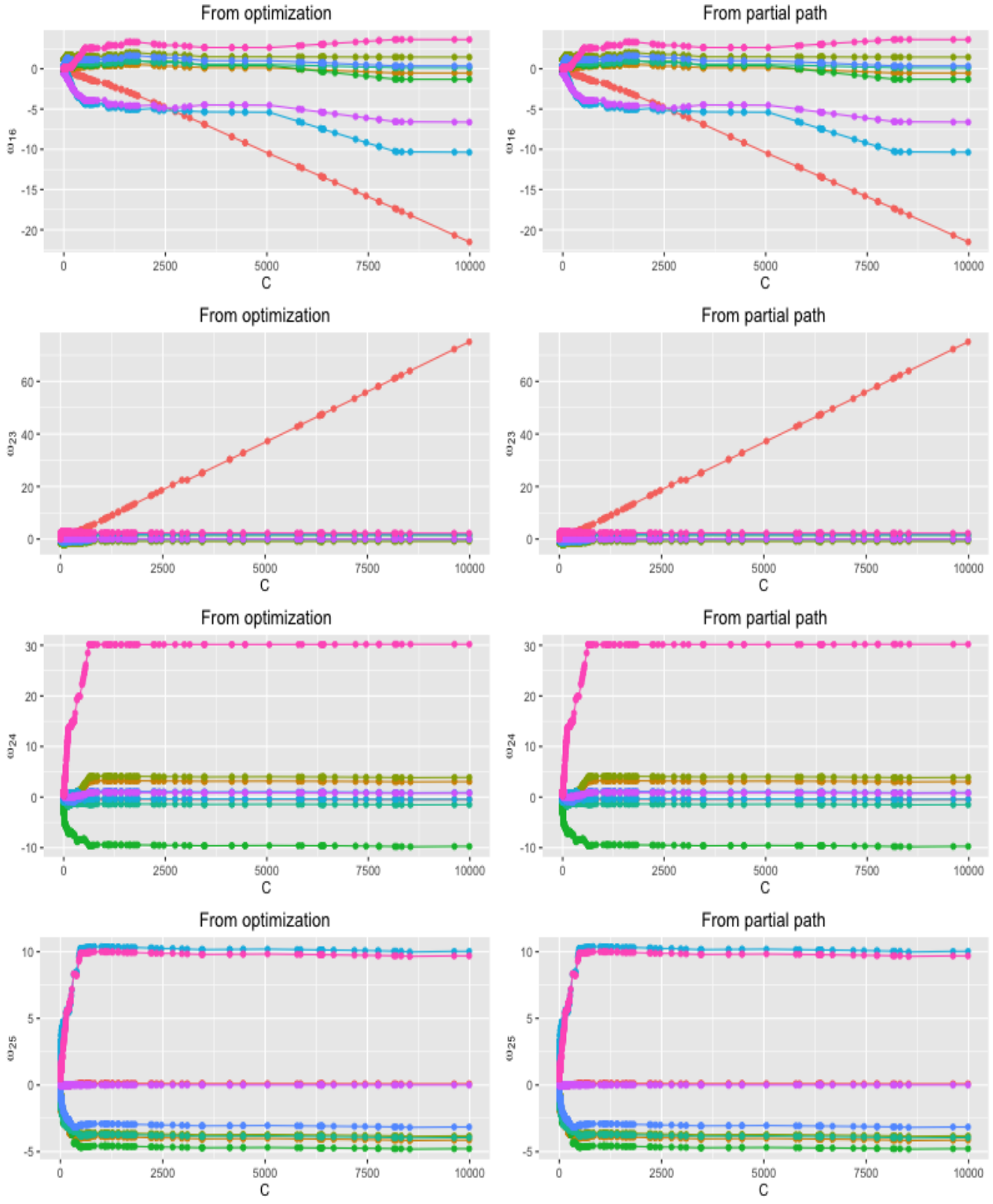


Figure B.7: Part3: Compare the solutions gotten from optimization and partial path algorithm for GLASS data with randomly chosen t_0, t_1

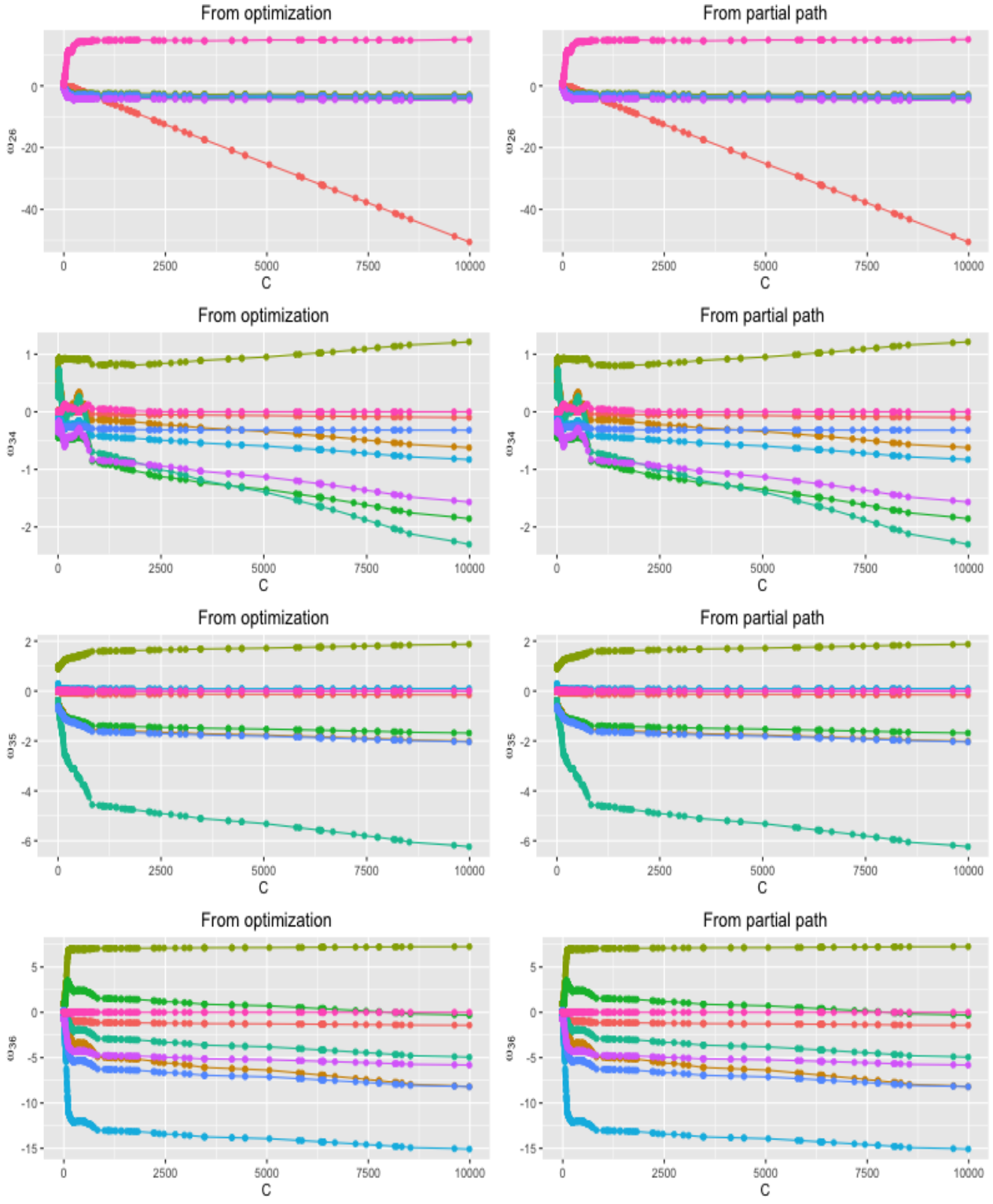
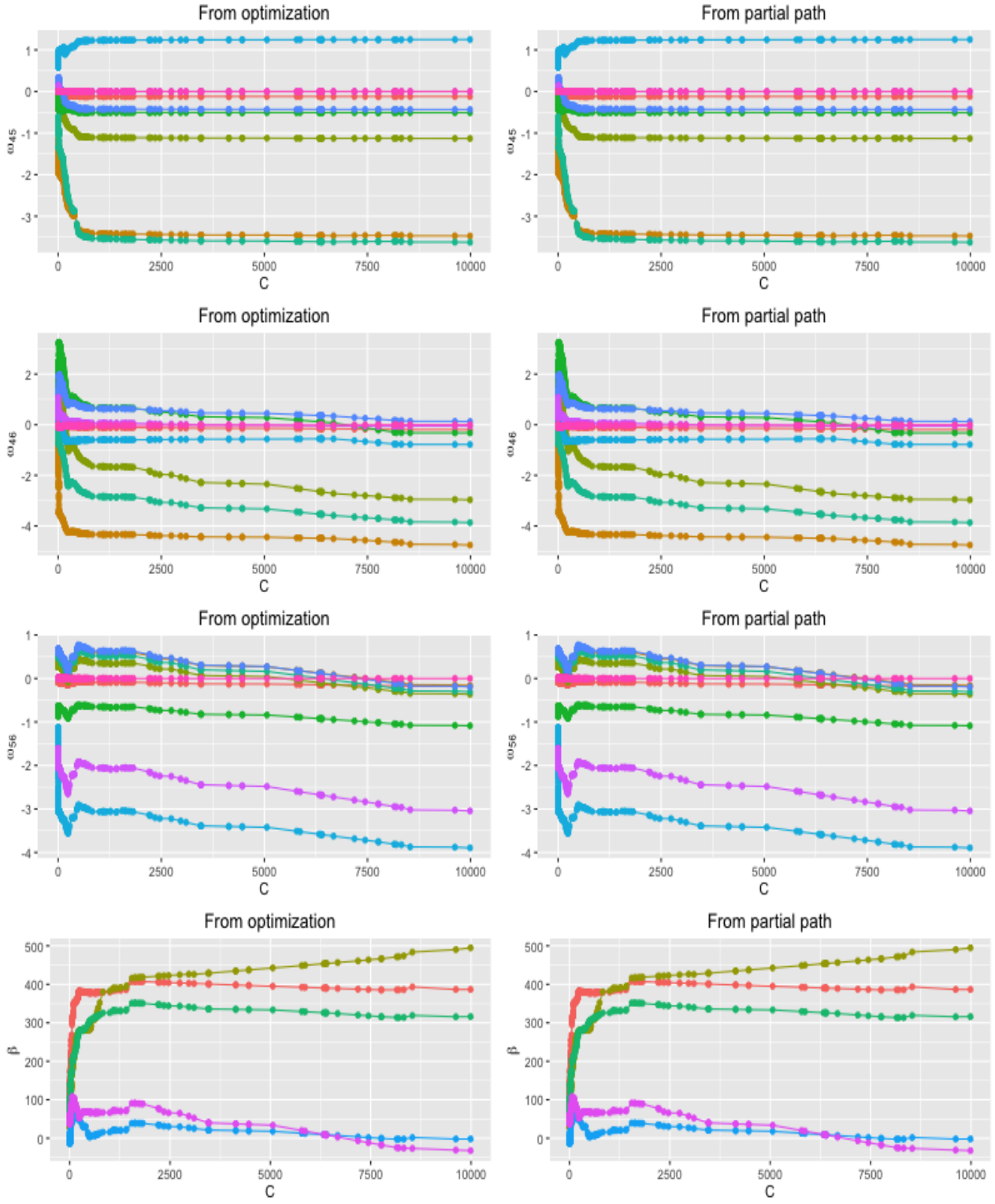


Figure B.8: Part4: Compare the solutions gotten from optimization and partial path algorithm for GLASS data with randomly chosen t_0, t_1



Bibliography

Akbani, R., S. Kwek, and N. Japkowicz (2004). Applying Support Vector Machines to Imbalanced Datasets. *Machine Learning: ECML 2004 3201*(July), 39–50.

Bach, F. R. (2006). Considering Cost Asymmetry in Learning Classifiers. *Jmlr 7*, 1713–1741.

Bahlmann, C., B. Haasdonk, and H. Burkhardt (2002). Online handwriting recognition with support vector machines - a kernel approach. In *Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition*, pp. 49–54. IEEE Comput. Soc.

Bredensteiner, E. and K. Bennett (1999). Multicategory classification by support vector machines. *Computational Optimization 12*(1-3), 53–79.

Carrizosa, E. and B. Martin-Barragan (2006). Two-group classification via a biobjective margin maximization model. *European Journal of Operational Research 173*(3), 746–761.

Chapelle, O., P. Haffner, and V. Vapnik (1999). Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks 10*(5), 1055–1064.

Chapelle, O., V. Vapnik, O. Bousquet, and S. Mukherjee (2002). Choosing multiple parameters for support vector machines. *Machine Learning 46*(1-3), 131–159.

Chinchuluun, A. and P. M. Pardalos (2007). A survey of recent developments in multiobjective optimization. *Annals of Operations Research 154*(1), 29–50.

Corne, D., C. Dhaenens, and L. Jourdan (2012). Synergies between operations research and data mining: {The} emerging use of multi-objective approaches. *European Journal of Operational Research 221*(3), 469–479.

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

Cover, T. and P. Hart (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory 13*(1), 21–27.

Crammer, K. and Y. Singer (2001). On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *Journal of Machine Learning Research 2*, 265–292.

Deb, K. (2001). Multi-objective optimization. *Multi-objective optimization using evolutionary ...*, 11–37.

Ding, C. H. and I. Dubchak (2001, apr). Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics 17*(4), 349–358.

Ehrgott, M. (2005). Multicriteria Optimization. *Lecture Notes in Economics and Mathematical Systems 5*, 595–600.

Fan, A. and M. Palaniswami (2001). Stock selection using support vector machines. *Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on 3*, 1793–1798 vol.3.

Fonseca, C. and J. Knowles (2005). A tutorial on the performance assessment of stochastic multiobjective optimizers. *... on Evolutionary Multi- ...*.

Friedrichs, F. and C. Igel (2005). Evolutionary tuning of multiple SVM parameters. *Neurocomputing 64*(Esann), 107–117.

Gendreau, M. and J.-Y. Potvin (2010). *Handbook of Metaheuristics*, Volume 146.

Guyon, I., J. Weston, S. Barnhill, and V. Vapnik (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 1–39.

Hansen, P. and N. Mladenović (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research 130*(3), 449–467.

Hansen, P., N. Mladenović, and J. Pérez (2010). Variable neighbourhood search: methods and applications. *Annals of Operations Research*.

Hastie, T., S. Rosset, R. Tibshirani, and J. Zhu (2004). The Entire Regularization Path for the Support Vector Machine. *Test 5*(2), 1391–1415.

Heisele, B., P. Ho, and T. Poggio (2001). Face recognition with support vector machines: global versus component-based approach. In *IEEE International Conference on Computer Vision, ICCV*, Volume 2, pp. 688–694.

Hsu, C.-W. and C.-J. Lin (2002). A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks 13*(2), 415–425.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *Machine Learning: ECML-98 1398*, 137 – 142.

Karasuyama, M., N. Harada, M. Sugiyama, and I. Takeuchi (2012). Multi-parametric solution-path algorithm for instance-weighted support vector machines. *Machine Learning 88*(3), 297–330.

Kim, K.-j. (2003). Financial time series forecasting using support vector machines. *Neurocomputing 55*(1-2), 307–319.

Kreßel, U. H.-G. (1999, feb). Pairwise classification and support vector machines. pp. 255–268.

Lifeng Wang, X. S. (2006). Multi-category support vector machines, feature selection and solution path. Vol.16, No.2.

Lin, H. and C. Lin (2003). A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods. *submitted to Neural Computation*, 1–32.

Lin, Y. (2002). Support vector machines and the Bayes rule in classification. *Data Mining and Knowledge Discovery 6*(3), 259–275.

Platt, J. (1999). Fast Training of Support Vector Machines Using Sequential Minimal Optimization. *Advances in kernel methods-support vector learning 3*.

Platt, J., N. Cristianini, and J. Shawe-Taylor (2000). Large Margin DAGs for Multiclass Classification. In *Advances in Neural Information Processing Systems*, pp. 547–553.

Rüping, S. (2001). SVM Kernels for Time Series Analysis. *Time*, 43–50.

Shoki Ishida, Keiji Tatsumi, T. T. (2012). A Multiobjective Multiclass Support Vector Machine Based on One-Against-One Method. In X. Qiu and H. Lau (Eds.), *Job shop scheduling with artificial immune systems*, pp. 75–78. Australia.

Tatsumi, K., M. Akao, R. Kawachi, and T. Tanino (2011). Performance evaluation of multiobjective multiclass support vector machines maximizing geometric margins. *Numerical Algebra, Control and Optimization 1*(1), 151–169.

Tatsumi, K., K. Hayashida, H. Higashi, and T. Tanino (2007). Multi-objective multiclass support vector machine for pattern recognition. *SICE Annual Conference 2007*, 1095–1098.

Tatsumi, K., R. Kawachi, K. Hayashida, and T. Tanino (2009a). Multiobjective multiclass soft-margin support vector machine and its solving technique based on benson's method. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 5861 LNAI*, 360–371.

Tatsumi, K., R. Kawachi, K. Hayashida, and T. Tanino (2009b). Multiobjective multiclass soft-margin support vector machine maximizing pair-wise interclass margins. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 5506 LNCS*(PART 1), 970–977.

Tatsumi, K., M. Tai, and T. Tanino (2010, jul). Multiobjective multiclass support vector machine based on the one-against-all method. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. IEEE.

Tatsumi, K., M. Tai, and T. Tanino (2011). Nonlinear extension of multiobjective multiclass support vector machine based on the one-against-all method. In *Proceedings of the International Joint Conference on Neural Networks*, pp. 1570–1576.

Tatsumi, K. and T. Tanino (2014, aug). Support vector machines maximizing geometric margins for multi-class classification. *TOP 22*(3), 815–840.

Tong, S. and D. Koller (2001). Support Vector Machine Active Learning with Applications to Text Classification. *Journal of Machine Learning Research*, 45–66.

Van Gestel, T., J. a. K. Suykens, D.-e. Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, and J. Vandewalle (2001). Financial Time Series Prediction Using Least Squares Support Vector Machines Within the Evidence Framework. *IEEE Transaction on Neural Networks 12*(4), 809–821.

Vapnik, V. and V. Vapnik (1982). *Estimation of Dependences Based on Empirical Data*.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*, Volume 8.

Vapnik, V. N. (1998). *Statistical Learning Theory*, Volume 2.

Veropoulos, K., C. Campbell, N. Cristianini, and Others (1999). Controlling the sensitivity of support vector machines. *Proceedings of the international joint conference on artificial intelligence*, 55–60.

Wang, L. and X. Shen (2007, jun). On L 1 -Norm Multiclass Support Vector Machines. *Journal of the American Statistical Association 102*(478), 583–594.

Weston, J. and C. Watkins (1998). Multi-class support vector machines. *Pattern Recognition*, 0–9.

Weston, J. and C. Watkins (1999). Support vector machines for multi-class pattern recognition. *Proceedings of the Seventh European Symposium On Artificial Neural Networks 4*, 6.

Zhang, H., A. C. Berg, M. Maire, and J. Malik (2006). SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2*, 2126–2136.

Zhu, J., S. Rosset, T. Hastie, and R. Tibshirani (2004). 1-norm Support Vector Machines. *Advances in Neural Information Processing Systems*, 49–56.

Zitzler, E., E. Zitzler, L. Thiele, M. Laumanns, C. M. A. F. C. M. Fonseca, V. G. A. d. F. V. G. da Fonseca, and M. Laumanns (2003). Performance assessment of multiobjective optimizers: an analysis and review. *Evolutionary Computation, IEEE Transactions on 7*(2), 117–132.