Advanced Learning Approaches Based on SVM+ Methodology

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Feng Cai

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Advisor: Vladimir S. Cherkassky

July, 2011

Acknowledgements

First of all, I would like to express my deepest thanks and appreciation to my academic advisor, Prof. Vladimir Cherkassky, who guided me through my PhD thesis research work. I sincerely appreciate his precious advice and consistent encouragement during my PhD study. It has been a pleasure and an honor working with him.

I would also like to thank my PhD committee members, Professor Jarvis Haupt, Professor Chad Myers and Professor Gerald Sobelman, for reviewing my thesis and giving valuable feedback and advice on my work.

I would like to thank Dr. Yanting Dong and Dr. Deepa Mahajan, who has been my mentors during my intern at Boston Scientific CRM, Dr. Mohammad Hasan and Nish Parikh, who has been my mentors during my intern at eBay research labs. Their experience and advice helped me deal with real life problems.

I would like to thank my colleagues in the machine learning lab, Wuyang Dai, Sauptik Dhar, Han-Tai Shiao, Sohini Roy Chowdhury, Tuo Zhao, Lichen Liang, Tao Xiong, Yunqian Ma and many others for their help and friendship.

Dedication

To my parents and my wife Xinran.

Abstract

Exploiting additional information to improve traditional inductive learning is an active research area in machine learning. In many supervised learning applications, training data contains additional information not reflected in training pairs $(\mathbf{x}, y)$. Examples include: (1) time series prediction where future samples can be observed in the training data, (2) handwritten digit recognition where training examples are provided by several persons, and this group information is not utilized during training, (3) medical diagnosis where predictive (diagnostic) model, say for lung cancer, is estimated using a training set of male and female patients. The gender can be considered as additional group information.

Incorporating this additional information into learning may improve generalization. Recently, Vapnik proposed a general approach for incorporating additional information into learning, known as Learning Using Privileged Information (LUPI) and learning with structured data (LWSD) which utilizes group information (Vapnik, 2006). A SVM based methodology SVM+ was proposed under LUPI and LWSD setting (Vapnik, 2006). In this thesis, we will first introduce a SVM+ based feature selection system. Then we extend SVM+ to multi-task learning (MTL) setting, where both training and test data can be naturally partitioned into several groups. SVM+ based MTL (SVM+MTL) method for both classification and regression are proposed and analyzed. SVM+MTL estimates multiple models simultaneously, i.e. one model for each group/task. Task inter-dependency is modeled by sharing a common part of the decision function among different groups. Connections and differences between SVM+ and SVM+MTL are discussed. Practical parameter tuning strategies are proposed for

SVM+MTL. Empirical comparisons show that SVM+MTL works very well on data sets with group information. Finally, generalized sequential minimal optimization (GSMO) methods are proposed for SVM+MTL training, for both classification and regression settings.

Table of Contents

List of Tables

List of Figures

Notation

The following notation is used throughout the thesis. Scalars are indicated by script letters such as a. Vectors are indicated by lowercase bold letters, such as $\mathbf{w}$. Matrices are given using uppercase bold letters $\mathbf{V}$. When elements of a matrix are accessed individually, we use the corresponding lowercase script letter. For example, the (i, j) element of the matrix $\mathbf{V}$ is v_{ij}.

n	Number of samples
d	Number of input variables, also dimensionality of the input/feature space
t	Number of different groups
$\mathbf{x}$	Input column data vector
y	Output of a learning system. For classification y is a binary label, for regression y is real-valued.
$\mathbf{X} = [\mathbf{x}_1, \mathrm{K}\ , \mathbf{x}_n]$	Matrix of input samples
$\mathbf{y} = [y_1, \mathrm{K}\ , y_n]$	Vector of output samples
$\mathbf{Z} = [\mathbf{z}_1, \mathrm{K}\ , \mathbf{z}_n]$	Representation of data points in a feature space
P	Cumulative Distribution Function
$p(\mathbf{x})$	Probability density function (pdf)
$p(y \mid \mathbf{x})$	Conditional density

$f(\mathbf{x}, \omega)$	A class of approximating functions indexed by abstract parameter ω (ω can be a scalar, vector or matrix)
$L(y, f(\mathbf{x}, \omega))$	Loss function
$(\mathbf{a} \cdot \mathbf{b}), \mathbf{a}^T\mathbf{b}$	Inner (dot) product of two vectors
$R(\omega)$	Expected risk as a function of parameters
$R_{emp}(\omega)$	Empirical risk as a function of parameters
$K(\mathbf{x}_1, \mathbf{x}_2)$	General kernel function
h	VC dimension
T_r	Set of indices of samples in group r
Δ	Margin distance
ξ	Error between the target function and the approximating function. Also used to denote additive noise for regression data.

In additional to the above notations, there are chapter-specific notations which will be introduced locally in each chapter.

Chapter 1 Introduction

1.1 Motivations

Learning is the process of estimating an unknown (input, output) dependency or structure of a system using a limited number of observations (Cherkassky and Mulier, 2007). Various applications in engineering, statistics, computer science, health sciences and social sciences are concerned with estimating 'good' predictive models from available historical data (or training data), in order to use this model for predicting future samples (or test data). Predictive learning is of particular interest because it can objectively define the 'usefulness' of the estimated model by predictive (generalization) capabilities.

A supervised learning system for single model estimation is shown in Figure 1.1 (Vapnik, 1995; Cherkassky and Mulier, 1998). Under standard (single) model formulation, we are given finite number of samples called training data $(\mathbf{x}_i, y_i), (i = 1, \mathrm{K}\ , n)$, and the goal of learning is to select the best predictive model $f(\mathbf{x}, \omega^*)$ from a set of functions $f(\mathbf{x}, \omega), \omega \in \Omega$ where Ω is a set abstract parameters used to index the set of functions. The selected model is expected to have good prediction accuracy for the future (test) data. The learning problem is called regression problem when the system's output is real-valued $(y \in R)$, and called classification problem when the system's output takes on a small number of discrete values. As a special case, it is called binary classification problem if $y \in \{+1, -1\}$.

Support Vector Machine (SVM) is a popular methodology for predictive learning with finite samples developed in VC-theory (Vapnik, 1982, 1985, 1998). Our new machine learning methodologies investigated in this thesis will be based on SVM. Conceptually, SVM utilizes a new form of parameterization of approximating functions, so that model complexity can be controlled independently of the problem dimensionality. Technical implementation of SVM incorporates several powerful ideas/concepts from the optimization theory (i.e. convex optimization) and functional analysis (i.e. Reproducing Kernels in Hilbert Space). In additional to its sound theoretical foundation, SVM has been shown to achieve great empirical success (Cherkassky and Mulier, 1998). SVM is less prone to overfitting due to its margin-based complexity control.

In this thesis, we focus on predictive learning on structured data. In a data-rich world, there often exist additional information about training samples, which is not reflected in the training set $(\mathbf{x}, y), (i = 1, \mathrm{K}\ , n)$. This additional information can be easily ignored by standard inductive methods such as SVM. Effective use of this additional information during training often results in improved generalization. Several examples illustrate this possibility:

- *Handwritten digit recognition*, where the training data is known to be generated by several persons. In this case, each training sample has an additional person (group) label; and this information can be used to improve generalization. However, this group label is not available during test, e.g., a classifier uses only pixels of an image for making prediction.

– *Medical diagnosis*, where the goal is to estimate a predictive classifier for diagnosing a disease based on the clinical tests, or input features $\mathbf{x}$, known at the time of initial examination. However, available patient data may include additional information, such as pathological reports and advanced tests performed later. This additional information (denoted as feature vector $\mathbf{x}^*$) can be used for training a classifier; however features $\mathbf{x}^*$ are not known during test (prediction), so the goal of learning is to estimate a decision rule $\hat{y} = f(\mathbf{x})$ using only input features known at the time of initial examination.

– *Medical diagnosis*, where the goal is to estimate a decision rule for diagnosing a disease using a number of clinical features $\mathbf{x}$. Usually, medical records also include demographic information (gender, age, and race) that does not seem relevant to diagnosis. For instance, patient's gender has no predictive value for diagnosing lung cancer. However, this demographic information may still be utilized for estimating a classifier $\hat{y} = f(\mathbf{x})$.

– *Brain imaging*. In fMRI data analysis, available data (fMRI images) originate from different subjects performing the same cognitive task. Then the goal may be to estimate a classifier to detect 'cognitive states' corresponding to these tasks. Such a classifier has to generalize well for new subjects, i.e. perform accurate classification of fMRI images. In this case, the training data can be naturally partitioned into several groups, where each group contains fMRI images from one subject.

- *Time series prediction*. Often the goal is to predict a future event based on past (known) values of a time series. For instance, we may want to predict whether a stock market index will be higher or lower in 30 minutes, using currently available information, such as the past values of this index and other input features. However, historical data (used for training) also includes the information about the values of input features during every 30-minute interval *before* the predicted time. This additional information can be incorporated into training, and may improve generalization.
- *Landmine detection*. The goal is to detect landmine based on a vector of feature inputs. The data is collected from different landmine fields. Data from one landmine field can be regarded as a group/task. The group information can be used to help improve prediction.

This additional information of training data brings new challenge to traditional inductive learning methods (aks single task learning) such as SVM. Effective use of this additional information in training data often leads to improved generalization. When additional information is not available in test data (as in *handwritten digit recognition* and *time series prediction* examples), traditional inductive learning methods cannot use the additional information. Therefore, new non-inductive methods are needed to utilize this additional information. The other case is that the additional information is group information and it is available during test (as in *landmine detection* examples). Since data from different groups may have different characteristics, single model estimated by traditional inductive learning, which ignores group information, may not have good

generalization performance. One straightforward way of incorporating group information into single model estimation is to encode this group information into a categorical variable and use this variable as an extra input. However, traditional single task learning (STL) method doesn't treat this extra input differently from other feature inputs. This group information is not effectively used by simply encoding it into an extra categorical input. This will be confirmed by the empirical results in chapter 4 and 5. Another strategy is to train multiple models independently, one model for each group. This will work when data from different groups are completely independent (not related to each other) and each group has enough number of samples. However, in many real applications, data from different groups tend to be related with each other and number of samples in each group is small. For example, in medical diagnosis where training data can be divided into two groups: male and female patients. The training data is collected from different patients. This data collection process is difficult as it takes many years and many patients don't want to participate in the study. In this case, training multiple models independently, which doesn't utilize all available training data effectively, is not a very promising idea.

There are three non-standard learning settings where additional information is effectively utilized to help improve generalization. One is Learning Using Privileged Information (LUPI) (Vapnik, 2006, 2009). Another one is Learning With Structured Data (LWSD) formulation (Vapnik, 2006). We will show that LWSD can be regarded as a special case of LUPI. The third one is Multi-Task Learning (MTL, aka Learning to Learning, Transfer Learning), main topic of our thesis.

In standard inductive learning (aka single task learning (STL)) setting (Vapnik, 1998; Cherkassky and Mulier, 2007; Hastie, T., 2001), we are given a training set $\mathbf{D} = \{(\mathbf{x}_i, y_i), (i = 1, \mathrm{K}\ , n)\}$ with samples identically and independently generated from an unknown probability distribution $p(\mathbf{x}, y)$. The goal is to find the best mapping function $f : \mathbf{x} \to y$ such that the expected loss

$$R_{STL}(\omega) = \iint_{X,Y} L(f(\mathbf{x}, \omega), y)\, p(\mathbf{x}, y) d\mathbf{x} dy \qquad (1.1)$$

is minimized. Here in (1.1) $L(f(\mathbf{x}, \omega), y)$ is used to denote a loss function. It can be 0/1 classification error (for classification problems), or squared error (for regression problems).

In LUPI setting, training data $(\mathbf{x}_i, \mathbf{x}_i^*, y_i), i = 1, \mathrm{K}\ , n$ is generated from an unknown probability distribution $p(\mathbf{x}, \mathbf{x}^*, y)$, where $(\mathbf{x}_i, y_i)$ is the 'usual' training data and $\mathbf{x}_i^*$ denotes additional privileged information. Note that this privileged information $\mathbf{x}_i^*$ is not available during test, as in *time series prediction* example. The goal is to find the best mapping $f : \mathbf{x} \to y$ such that the expected loss

$$R_{LUPI}(\omega) = \iint_{X,Y} L(f(\mathbf{x}, \omega), y)\, p(\mathbf{x}, \mathbf{x}^*, y) d\mathbf{x} dy$$

is minimized.

Now suppose the training data comes from t related groups. For each group r, we have n_r data points sampled from a distribution p_r on $\mathbf{X} \times \mathbf{Y}$. So the data is a union of $t > 1$ related groups: $\{\{X_r, Y_r\}, r = 1, \mathrm{K}\ , t\}, \{X_r, Y_r\} = \{\{\mathbf{x}_{r_1}, y_{r_1}\}, \mathrm{K}\ , \{\mathbf{x}_{r_{n_r}}, y_{r_{n_r}}\}\}$ and can

be thought as samples identically and independently generated from the distribution $p = \cup_{r=1,\mathrm{K},t} p_r$.

If the group labels of future test samples are not given, as in *handwritten digit recognition* example, the setting is LWSD and the goal is to find one best mapping function f such that the expected loss

$$R_{LWSD}(\omega) = \iint_{X,Y} L(f(\mathbf{x}, \omega), y)\, p(\mathbf{x}, y) d\mathbf{x} dy \qquad (1.2)$$

is minimized. The difference between (1.1) and (1.2) is that p in (1.1) is an unknown probability while p in (1.2) is known to be a union of t sub-distributions. Note that LWSD can be regarded as a special case of LUPI. The group information in LWSD can be encoded into an extra categorical variable and this extra information can be used as privilege information $\mathbf{x}^*$ in LUPI. However, under LWSD, we have more flexible manipulation of group information (i.e. each group has a unique correcting function, to be discussed later).

On the other hand, if the group labels of future test samples are given, as in *medical diagnosis* example, the setting is MTL and the goal is to find t mapping functions $\{f_1, f_2, \mathrm{K}, f_t\}$ such that the sum of expected loss for each task

$$R_{MTL}(\omega) = \sum_{r=1}^{t} \iint_{X,Y} L(f_r(\mathbf{x}, \omega), y) P_r(\mathbf{x}, y) d\mathbf{x} dy \qquad (1.3)$$

is minimized. Different ways of handling group information by several methods are shown in Figure 1.2.

LWSD and MTL settings are similar in the sense that both of them try to utilize group information during training, as available training data is naturally segmented into several groups. However, the difference is that for MTL the group (task) label is also known for test data. So under MTL the goal is to estimate several related models, whereas LWSD estimates a single model.

Recently, a SVM based method called SVM+ was developed under both LUPI setting (Vapnik, 2006, 2009) and LWSD setting (Vapnik, 2006). SVM+ introduces several new concepts such as correcting function (to be discussed later) and projecting training data into two different spaces. These new ideas have lead to the SVM+ based MTL (SVM+MTL) method for classification (Liang and Cherkassky, 2008). We empirically compared SVM+MTL classification with several methods such as SVM, SVM+ and regularized Multi-Task Learning (rMTL) (Evgeniou and Pontil, 2004). In many real applications we need to deal with unequal misclassification cost. For example, in Statlog heart disease classification, cost of false positive and false negative are different (Michie et. al., 1994). We adapted the SVM+MTL classification algorithm to deal with unequal misclassification cost. We then proposed SVM+MTL algorithm for regression. The SVM+MTL algorithm introduces more tuning parameters (to be discussed later), practical parameter tuning/model selection strategies are designed.

The training algorithm of SVM needs to solve a large Quadratic Programming (QP) problem. The computational complexity is $O(n^3)$, where n is sample size of training data (Tsang et al., 2005). This is slow when n is large. A common method for solving the QP problem in SVM training is Sequential Minimal Optimization (SMO)

algorithm (Platt, 1998), which breaks the problem down into 2-dimensional sub-problems that may be solved analytically, eliminating the need for a numerical optimization problem. A similar QP problem needs to be solved in order to train SVM+MTL. We designed the Generalized SMO (GSMO) for both SVM+MTL classification and SVM+MTL regression. Empirical results show that the computational complexity of GSMO is about $O(n^{2.2})$.

1.2 Literature Review for Multi-Task Learning

Learning multiple related tasks simultaneously often results in improved generalization, relative to learning each task independently. Various constructive MTL approaches have been developed in machine learning. All these approaches provide different formalization of the 'task-relatedness'. They can be broadly grouped into four categories.

One category of MTL is to learn small sets of relevant features that are shared across tasks (Obozinski, Taskar and Jordan, 2006; Argyriou, Evgeniou and Pontil, 2006). These methods are the so called Multi-Task Feature Selection/Learning methods. Obozinki et al. proposed a joint regularization framework that extends l_1 regularization for a single task to a MTL setting by penalizing the sum of l_2-norms of the block of coefficients associated with each feature across tasks. This can be expressed as a joint optimization that combines l_1 and l_2 norms on the coefficients matrix. Argyriou et al. extends the formulation of Obozinki et al. by introducing an intermediate hidden representation.

Another category is to learn multiple related models by sharing common internal representation (Bakker and Heskes, 2003; Ando and Zhang, 2005; Liao and Carin, 2005). These methods are generalized from neural networks, where multiple-output MLP network can model several (related) classification or regression tasks, which share the same hidden units. Other MTL methods, which model 'task relatedness' by sharing common priors among different tasks (Raina, Ng and Koller, 2006; Lawrence and Platt, 2004; Xue, Liao, Carin and Krishnapuram, 2007), fall into the third category.

The last category of methods learn multiple models through kernel methods and regularization (Evgeniou and Pontil, 2004; Liang and Cherkassky 2008). Evgeniou et al. proposed regularized Multi-Task Learning (rMTL), where models of different tasks share a common part. Kernel methods are used in rMTL so that the inputs are projected into a feature space. Liang et al. proposed a similar MTL method based on SVM (SVM+MTL) for classification problems. The difference is that SVM+MTL projects data into two different spaces, which is introduced by SVM+. Our thesis will focus on SVM+MTL methods for both classification and regression problems.

1.3 Structure of the Dissertation

The rest of the thesis is organized as follows:

Chapter 2 introduces background for predictive learning. Then, SVM for both classification and regression are reviewed.

Chapter 3 introduces SVM+ based feature selection learning system. Features (not available during test) removed by feature selection can be used by SVM+ as privileged information to improve generalization.

Chapter 4 describes SVM+ and SVM+MTL for classification. SVM+MTL is adapted to handle unequal misclassification costs. Empirical comparisons are made between SVM+MTL and several other methods such as SVM, SVM+ and rMTL.

Chapter 5 presents SVM+MTL for regression. Practical parameter tuning strategies are introduced.

Chapter 6 proposes GSMO for both SVM+MTL classification and SVM+MTL regression.

Chapter 7 gives summary.

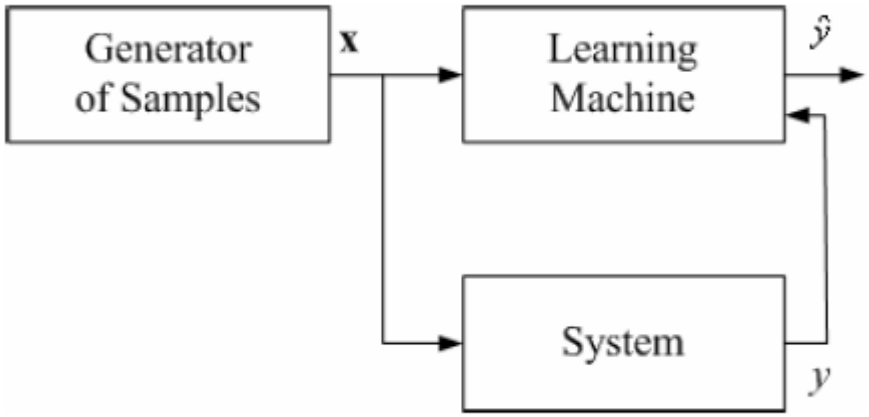

Figure 1.1: Learning system for single model estimation. During the learning process, the learning machine observes pairs $(\mathbf{x}, y)$ (training set). After training, for any given input $\mathbf{x}$, the learning machine returns a value $\hat{y}$. The goal of learning is to generate $\hat{y}$ values that imitate the system's output y.

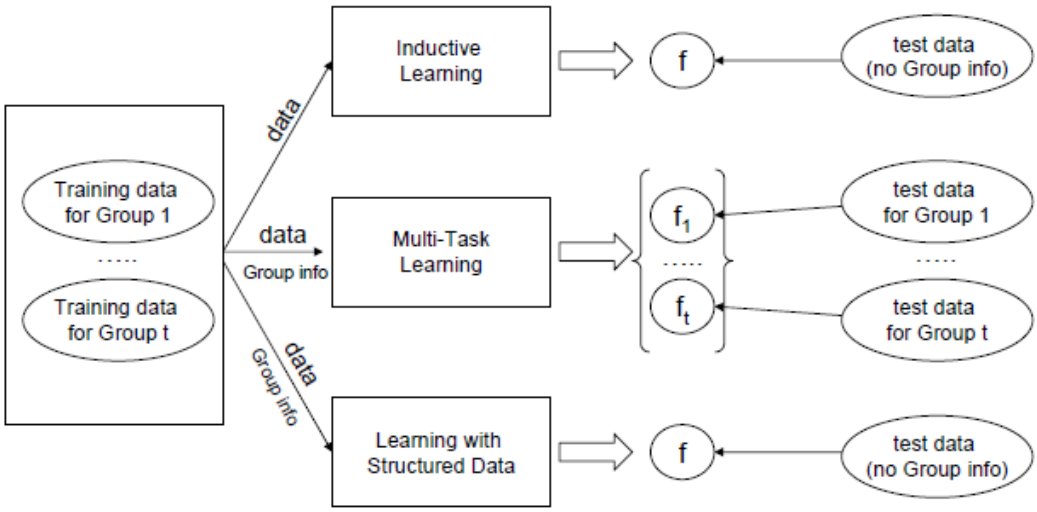

Figure 1.2 Inductive learning, Multi-Task Learning and Learning With Structured Data handle training and test data in different ways.

Chapter 2 Background on Predictive Learning

This chapter reviews the basic concepts of predictive learning and SVM, following (Cherkassky and Mulier, 2007).

2.1 Objective of Predictive Learning

Learning is the process of estimating an unknown (input, output) dependency or structure of a system using a limited number of observations. The finite training set is denoted by $\mathbf{D} = \{(\mathbf{x}_i, y_i), (i = 1, \mathrm{K}\ , n)\}$ with samples identically and independently generated from an unknown probability distribution $P(\mathbf{x}, y)$. Suppose the set of functions estimated by the learning machine is denoted by $f(\mathbf{x}, \omega)$, where ω is the index. The quality of an approximation produced by the learning machine is measured by the loss $L(f(\mathbf{x}, \omega), y)$ or discrepancy between the output y produced by the original system and the output $f(\mathbf{x}, \omega)$ produced by learning machine for a given input $\mathbf{x}$. The expected value of the loss is called the *risk functional*:

$$R_{STL}(\omega) = \iint_{X,Y} L(f(\mathbf{x}, \omega), y)\, p(\mathbf{x}, y) d\mathbf{x} dy .$$

The goal of learning to estimate the function $f(\mathbf{x}, \omega_0)$, which minimizes the risk functional over the set of functions supported by the learning machine using only the training data. With finite data we cannot expect to find $f(\mathbf{x}, \omega_0)$ exactly, so we denote $f(\mathbf{x}, \omega^*)$ as the estimate of the optimal solution obtained with finite training data using

some learning procedure. The estimated function $f(\mathbf{x}, \omega^*)$ is expected to have good prediction accuracy for future (test) data. In practice, in order to evaluate a machine learning algorithm or compare different algorithms, we use the so called test error as criteria. Suppose we are given a test set $(\mathbf{x}_j, y_j), (j = 1, \mathrm{K}, m)$, the test error is defined as

$$\mathrm{Error}_{\mathrm{test}} = \sum_{j=1}^{m} L(f(\mathbf{x}_j, \omega), y_j). \qquad (2.1)$$

2.2 Classification and Regression

Classification and regression are two common learning tasks under the generic learning problem. A general description is given here as these two tasks will be used in the thesis. The differences between classification and regression are the outputs and loss functions.

2.2.1 Classification

In this thesis, we mainly deal with binary classification. The output of binary classification system takes on only two values $y = \{+1, -1\}$ corresponding to two classes. Therefore, in learning machine, $f(\mathbf{x}, \omega), \omega \in \Omega$ are a set of indicator functions. A commonly used loss function for binary classification problem is the misclassification error:

$$L(f(\mathbf{x}, \omega), y) = \begin{cases} 0 & \text{if } y = f(\mathbf{x}, \omega), \\ 1 & \text{if } y \neq f(\mathbf{x}, \omega). \end{cases}$$

2.2.1 Regression

The output of the system in regression problem is a variable that takes on real values: $y \in R$. In learning machine, $f(\mathbf{x}, \omega), \omega \in \Omega$ are a set of functions with real values. A common loss function for regression is the squared error:

$$L(f(\mathbf{x}, \omega), y) = (y - f(\mathbf{x}, \omega))^2$$

2.3 Support Vector Machine (SVM)

In this section, we will review SVM, as our advanced algorithms in this thesis are based on SVM. According to VC theory, the generalization bound for learning with finite samples is as follows:

$$R(\mathbf{w}) \leq R_{\text{emp}}(\mathbf{w}) + \Phi(R_{\text{emp}}(\mathbf{w}), h/n, -\ln\eta/n). \qquad (2.2)$$

Detailed analysis suggests that the second term (confidence interval Φ) depends mainly on the VC dimension (or the ratio h/n), whereas the first term (empirical risk) depends on parameters $\mathbf{w}$. SVM provides a special way to achieve small empirical risk (first term) using low complexity (second term) parameterizations. Therefore, SVM provides good generalization for future (test) data.

2.3.1 SVM for Classification

Let's consider binary classification setting where we are given finite training data $(\mathbf{x}_i, y_i), i = 1,\text{K}\ ,n,\ \ \mathbf{x} \in R^d$ and $y \in \{+1, -1\}$. The goal of SVM is to find the optimal decision function $D(\mathbf{x}) = \mathbf{w}\mathbf{x} + b$ with good generalization performance.

Assuming that training data is linearly separable, there are many separating hyperplanes ($g(\mathbf{x}) = \mathbf{w}g\mathbf{x} + b$) satisfying the constraints

$$y_i(\mathbf{w}g\mathbf{x}_i + b) \geq 1, i = 1,\text{K}\ , n.$$

SVM approach considers an optimal separating hyperplane (Vapnik 1995), for which the margin (the distance between the closest data points to the hyperplane) is maximized. The VC dimension (model complexity) of an optimal separating hyperplane is

$$h \leq \min(\frac{r^2}{\text{Margin}^2}, d) + 1$$

where r is the radius of the smallest sphere that contains the training input vectors $(\mathbf{x}_1,\text{K}\ ,\mathbf{x}_n)$. SVM implements structural risk minimization (SRM) inductive principle by keeping the value of empirical risk fixed (i.e. zero for linearly separable case) and minimizing the confidence interval (by maximizing margin). The concept of margin is illustrated in Figure 2.1. Maximization of margin is equivalent to minimization of $\| \mathbf{w} \|$. To this end, SVM solves the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \frac{1}{2} \| \mathbf{w} \|^2 \\ \text{subject to:} \quad & y_i(\mathbf{w}g\mathbf{x}_i + b) \geq 1,\ i = 1,\text{K}\ , n. \end{aligned} \tag{2.3}$$

When training data is not linearly separable, the empirical risk $R_{\text{emp}}(\mathbf{w})$ is no longer zero. That is, some training samples are allowed to fall inside the margin (so called soft margin). Non-negative slack variables $\xi_i = \max(1 - y_i f(\mathbf{w}, \mathbf{x}_i), 0), i = 1,\text{K}\ , n$, which represent deviations from the margin borders, are introduced (Figure 2.2).

Empirical risk is then defined as: $R_{\text{emp}}(\mathbf{w}) = \sum_{i=1}^{n} \xi_i$. In this case, SVM attempts to strike a balance between the goal of minimization of empirical risk and maximizing the margin:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2} \| \mathbf{w} \|^2 + C \sum_{i=1}^{n} \xi_i \\ \text{subject to:} \quad & y_i (\mathbf{w} \mathbf{g} \mathbf{x}_i + b) \geq 1 - \xi_i, \ i = 1, \text{K} \ , n. \end{aligned} \tag{2.4}$$

In this form, the coefficient C controls the trade-off between complexity and proportion of non-separable samples and must be determined by model selection.

Problem (2.4) is a quadratic programming (QP) problem, it is common that it is solved by solving its dual form (according to convex optimization, the primal and dual forms of QP are equivalent):

$$\begin{aligned} \min_{\alpha} \quad & -\sum_{i=1}^{n} \alpha_i + \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to:} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \ i = 1, \text{K} \ , n. \end{aligned} \tag{2.5}$$

Formulation (2.5) has the solution in the form

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i (\mathbf{x} \mathbf{g} \mathbf{x}_i) + b \tag{2.6}$$

In the expansion (2.6), the sample points with non-zero α_i are called support vectors (SVs). The bias term b is given by

$$b = y_s - \sum_{i=1}^{n} \alpha_i y_i (\mathbf{x}_i \mathbf{g} \mathbf{x}_s)$$

where $(\mathbf{x}_s, y_s)$ is one of the support vectors.

2.3.2 SVM for Regression

In the case of regression, an appropriate margin-based loss function is ε-insensitive loss $L_\varepsilon(f(\mathbf{x},\mathbf{w}),y) = \max(|y - f(\mathbf{x},\mathbf{w})| - \varepsilon, 0)$ (Figure 2.3). The ε-insensitive loss can be formally described by introducing non-negative slack variables $\xi_i, \xi_i^*, i = 1,\text{K}, n$ to measure the deviation of training samples outside ε-insensitive zone (Figure 2.4). The empirical risk can then be expressed as sum of the slack variables: $R_{\text{emp}}(\mathbf{w}) = \sum_{i=1}^{n}(\xi_i + \xi_i^*)$.

SVM regression attempts to strike a balance between minimization of empirical risk and the penalization term:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ \text{subject to:} \quad & y_i - (\mathbf{w}\cdot\mathbf{x}_i) - b \le \varepsilon + \xi_i \\ & (\mathbf{w}\cdot\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \ge 0,\ i = 1,\text{K}, n. \end{aligned} \qquad (2.7)$$

The parameter C controls the trade-off between the empirical risk and the penalization term. In practice, an optimal value of C can be determined based on the range of response (y) values (Cherkassky and Ma, 2004). Parameter ε controls the model complexity (size of margin) and is tuned through model selection.

Problem (2.7) is usually solved by solving it dual form:

$$\begin{aligned} \min_{\alpha,\beta} \quad & \varepsilon\sum_{i=1}^{n}(\alpha_i + \beta_i) - \sum_{i=1}^{n} y_i(\alpha_i - \beta_i) + \frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \beta_i)(\alpha_j - \beta_j)(\mathbf{x}_i\cdot\mathbf{x}_j) \\ \text{subject to:} \quad & \sum_{i=1}^{n}\alpha_i = \sum_{i=1}^{n}\beta_i \\ & 0 \le \alpha_i \le C,\ 0 \le \beta_i \le C,\ i = 1,\text{K}, n. \end{aligned} \qquad (2.8)$$

The values of parameters $\alpha_i, \beta_i, i = 1, \mathrm{K}, n$ found by solving problem (2.8) give the SVM regression function:

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\alpha_i - \beta_i)(\mathbf{x}_i \mathbf{g} \mathbf{x}) + b \qquad (2.9)$$

The samples with non-zero coefficients are support vectors (SVs), corresponding to data points at or outside ε-insensitive zone. The bias term b is given by

$$b = y_s - \sum_{i=1}^{n} (\alpha_i - \beta_i)(\mathbf{x}_i \mathbf{g} \mathbf{x}_s)$$

where $(\mathbf{x}_s, y_s)$ is any support vector.

2.3.2 Nonlinear SVM

The linear SVM can be extended to nonlinear support vector classification/regression by mapping input vectors $\mathbf{x}$ into a high dimensional feature space Z through some nonlinear mapping $\mathbf{z} = \phi(\mathbf{x})$, and optimal separating hyperplane is constructed in the feature space (Vapnik, 1995). Note that we do not need to specify $\mathbf{z} = \phi(\mathbf{x})$ explicitly. The only thing we need to do is to replace the dot product $(\mathbf{x}_i \mathbf{g} \mathbf{x}_j)$ (in formulations (2.5),(2.6),(2.8) and (2.9)) with some kernel $K(\mathbf{x}_i, \mathbf{x}_j)$. Commonly used kernel functions include:

- Polynomial kernel (of degree q)

$$K(\mathbf{x}_i, \mathbf{x}_j) = ((\mathbf{x}_i \mathbf{g} \mathbf{x}_j) + 1)^q$$

- Radial Basis Function (RBF) kernel (with width parameter σ)

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{\| \mathbf{x}_i - \mathbf{x}_j \|^2}{2\sigma^2})$$

2.4 SVM+

This section reviews SVM+, a SVM based non-inductive methodology proposed by Vapnik, under two different settings: Learning Using Privileged Information (LUPI) and Learning With Structured Data (LWSD) (Vapnik, 1982-2006). In a data-rich world, there often exist additional information about training samples, which is not reflected in the training set $(\mathbf{x}, y), (i = 1, \mathrm{K}\ , n)$. This additional information can be easily ignored by standard inductive methods such as SVM. Effective use of this additional information during training often results in improved generalization. Several examples illustrate this possibility:

- *Handwritten digit recognition*, where the training data is known to be generated by several persons. In this case, each training sample has an additional person (group) label; and this information can be used to improve generalization. However, this group label is not available during test, e.g., a classifier uses only pixels of an image for making prediction.
- *Medical diagnosis*, where the goal is to estimate a predictive classifier for diagnosing a disease based on the clinical tests, or input features $\mathbf{x}$, known at the time of initial examination. However, available patient data may include additional information, such as pathological reports and advanced tests performed later. This additional information (denoted as feature vector $\mathbf{x}^*$) can be used for training a classifier; however features $\mathbf{x}^*$ are not known during test (prediction),

so the goal of learning is to estimate a decision rule $\hat{y} = f(\mathbf{x})$ using only input features known at the time of initial examination.

– *Medical diagnosis*, where the goal is to estimate a decision rule for diagnosing a disease using a number of clinical features $\mathbf{x}$. Usually, medical records also include demographic information (gender, age, and race) that does not seem relevant to diagnosis. For instance, patient's gender has no predictive value for diagnosing lung cancer. However, this demographic information may still be utilized for estimating a classifier $\hat{y} = f(\mathbf{x})$.

– *Brain imaging*. In fMRI data analysis, available data (fMRI images) originate from different subjects performing the same cognitive task. Then the goal may be to estimate a classifier to detect 'cognitive states' corresponding to these tasks. Such a classifier has to generalize well for new subjects, i.e. perform accurate classification of fMRI images. In this case, the training data can be naturally partitioned into several groups, where each group contains fMRI images from one subject.

– *Time series prediction*. Often the goal is to predict a future event based on past (known) values of a time series. For instance, we may want to predict whether a stock market index will be higher or lower in 30 minutes, using currently available information, such as the past values of this index and other input features. However, historical data (used for training) also includes the information about the values of input features during every 30-minute interval *before* the

predicted time. This additional information can be incorporated into training, and may improve generalization.

– *Landmine detection*. The goal is to detect landmine based on a vector of feature inputs. The data is collected from different landmine fields. Data from one landmine field can be regarded as a group/task. The group information can be used to help improve prediction.

This additional information of training data brings new challenge to traditional inductive learning methods (aks single task learning) such as SVM. Effective use of this additional information in training data often leads to improved generalization. When the additional information is not available in test data (as in *handwritten digit recognition* and *time series prediction* examples), two different learning setting, LUPI and LWSD, is proposed by Vapnik to utilize the additional information during training (Vapnik, 1982-2006).

2.4.1 SVM+ under LUPI setting

Under LUPI setting, we are given a set of triplets $(\mathbf{x}_i, \mathbf{x}_i^*, y_i), \mathbf{x}_i \in X, \mathbf{x}_i^* \in X^*, y_i \in \{+1, -1\}, i = 1, \text{K}, n$ generated according to a fixed but unknown probability $p(\mathbf{x}, \mathbf{x}^*, y)$, where $(\mathbf{x}_i, y_i)$ is the 'usual' training data and $(\mathbf{x}_i^*)$ denotes additional *privileged* information. The goal is to find among a given set of functions $f(\mathbf{x}, \omega), \omega \in \Omega$ the optimal function $f(\mathbf{x}, \omega^*)$ which guarantees the smallest probability of incorrect classification. According to VC theory, the new setting requires construction of a new SRM structure on the training set. Under LUPI setting, additional privileged information is specified in a different feature space, but this information is

somehow related to errors in the input feature space. Recently, a new technology called SVM+ has been developed by Vapnik (1982-2006) for learning under LUPI setting. This SVM+ approach maps inputs $\mathbf{x}_i, \mathbf{x}_i^*$ into two different spaces:

- *decision space* $\mathbf{Z}$ using mapping $\Phi(\mathbf{x}): \mathbf{x} \to \mathbf{Z}$, where the decision function $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{z} + b$ needs to be estimated. This is the same feature space as used in standard SVM;

- *correcting space* $\mathbf{Z}^*$ via mapping $\Phi^*(\mathbf{x}^*): \mathbf{x}^* \to \mathbf{Z}^*$, which reflects privileged information about training data. The correcting function $f^*(\mathbf{x}^*) = \mathbf{w}^* \cdot \mathbf{z}^* + d$ is defined in this space. This correcting function $f^*(\mathbf{x}^*)$ puts additional constraints on the training errors, or slack variables, in the decision space.

To this end, SVM+ estimates the decision function $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{z} + b$ by solving the following problem:

$$\begin{aligned} \min_{\mathbf{w},b,\mathbf{w}^*,d} \quad & \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + \frac{\gamma}{2}(\mathbf{w}^* \cdot \mathbf{w}^*) + C\sum_{i=1}^{n} \xi_i \\ \text{subject to:} \quad & y_i(\mathbf{w} \cdot \mathbf{z}_i + b) \geq 1 - \xi_i, \; i = 1, \ldots, n \\ & \xi_i = \mathbf{w}^* \cdot \mathbf{z}^* + d \geq 0, \; i = 1, \ldots, n \end{aligned} \qquad (2.10)$$

where $C > 0$ and $\gamma > 0$ are hyperparameters, $\mathbf{z}$ and $\mathbf{z}^*$ are feature maps of $\mathbf{x}$ and $\mathbf{x}^*$. The term $\gamma(\mathbf{w}^* \cdot \mathbf{w}^*)/2$ is intended to restrict the capacity (or VC-dimension) of the correcting function.

A common approach to solve (2.10) is to consider its dual problem:

$$\min_{\alpha,\beta} \; -\sum_{i=1}^{n}\alpha_i + \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j (\mathbf{z}_i \cdot \mathbf{z}_j) + \frac{1}{2\gamma}\sum_{i,j=1}^{n}(\alpha_i+\beta_i-C)(\alpha_j+\beta_j-C)(\mathbf{z}_i^* \cdot \mathbf{z}_j^*) \qquad (2.11)$$

$$\text{subject to: } \sum_{i=1}^{n}(\alpha_i+\beta_i-C)=0,\ \sum_{i=1}^{n} y_i\alpha_i = 0,$$

$$\alpha_i \geq 0, \beta_i \geq 0,\ i = 1,\text{K}\ ,n.$$

2.4.2 SVM+ under LWSD setting

Now we suppose the additional information is group information, as in *Brain Imaging* and *Landmine Detection* examples. Formally, under LWSD setting, the training data comes from t related groups. For each group r, we have n_r data points sampled from a distribution P_r on $\mathbf{X}\times\mathbf{Y}$. So the data is a union of $t>1$ related groups: $\{\{X_r, Y_r\}, r = 1,\text{K}\ ,t\}, \{X_r, Y_r\} = \{\{\mathbf{x}_{r_1}, y_{r_1}\},\text{K}\ ,\{\mathbf{x}_{r_{n_r}}, y_{r_{n_r}}\}\}$ and can be thought as samples identically and independently generated from the distribution $P = \cup_{r=1,\text{K}\ ,t} P_r$. The goal is to find one best mapping function f such that the expected loss

$$R_{LWSD}(\mathbf{w}) = \iint_{X,Y} L(f(\mathbf{x},\mathbf{w}), y)P(\mathbf{x}, y)d\mathbf{x}dy$$

is minimized.

In order to be useful for learning, this group information needs to be related to training errors, or slack variables ξ_i used in the standard soft-margin SVM formulation. In standard SVM, the only constraint on ξ_i is that it must be non-negative. The group information can be used to introduce additional constraints on the slack variables (errors) for samples from different groups. That is, we need to introduce *different constraints on*

slacks from different groups. This leads to the main idea of SVM+ under LWSD setting, where the slack variables for each group are modeled by the so-called *correcting functions*:

$$\xi_i^r = \xi_r(\mathbf{x}_i) = \phi_r(\mathbf{x}_i, \mathbf{w}_r),\ i \in T_r, r = 1,...,t. \qquad (2.12)$$

These correcting functions $\xi_r(\mathbf{x}_i) = \varphi_r(\mathbf{x}_i, \mathbf{w}_r)$ for group T_r, are defined in the correcting space $\mathbf{Z}_r$. That is, the input vectors $\mathbf{x}_i, i \in T_r$ are mapped into two different spaces:

- the decision space $\mathbf{Z}$ using mapping $\Phi(\mathbf{x}): \mathbf{x} \to \mathbf{Z}$ (as in standard SVM), and
- the correcting space $\mathbf{Z}_r$ via mapping $\Phi_{Z_r}(\mathbf{x}): \mathbf{x} \to \mathbf{Z}_r$. The correcting functions for different groups are specified in $\mathbf{Z}_r$ space.

Figure 2.5 illustrates this SVM+ mapping for the two groups in the training data $(t = 2)$. Even though Fig. 2.5 shows linear parameterization in the decision and correcting spaces; the mappings $\Phi(\mathbf{x}), \Phi_{Z_r}(\mathbf{x})$ themselves may be nonlinear.

Consider linear parameterization for the correcting functions:

$$\xi_r(\mathbf{x}_i) = (\mathbf{w}_r \cdot \mathbf{z}_i^r) + d_r \geq 0,\ r = \{1,...,t\} \qquad (2.13)$$

Here $\mathbf{w}_r, d_r$ denote the set of parameters and the bias term for the linear model for slacks from group r. These correcting functions provide additional constraints on the slack variables (errors) in the decision space. The correcting functions (2.13) are non-negative because they correspond to slack variables (as in standard SVM). So SVM+ learning pursues several objectives:

- *(SVM+1)* Separate labeled training data using large-margin hyperplane in the decision space $\mathbf{Z}$, as in standard inductive SVM;

- *(SVM+2)* Incorporate group information, in the form of additional constraints on the slack variables. These constraints are modeled as correcting functions in the correcting space $\mathbf{Z}_r$.
- *(SVM+3)* Control the capacity of a set of correcting functions, e.g. by restricting the norm $(\mathbf{w}_r \cdot \mathbf{w}_r)$.

Combining these goals leads to the following SVM+ optimization formulation:

$$\begin{aligned} \min_{\substack{\mathbf{w},b,\mathbf{w}_r,d_r \\ r=1,\mathrm{K},t}} \quad & \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + \frac{\gamma}{2}\sum_{r=1}^{t}(\mathbf{w}_r\cdot\mathbf{w}_r) + C\sum_{r=1}^{t}\sum_{i\in T_r}\xi_i^r \\ \text{subject to:} \quad & y_i((\mathbf{w}\cdot\mathbf{z}_i)+b) \geq 1-\xi_i^r, \\ & \xi_i^r = (\mathbf{w}_r\cdot\mathbf{z}_i)+d_r \geq 0,\ i\in T_r,\ r=1,\mathrm{K},t. \end{aligned} \qquad (2.14)$$

The solution to this optimization problem is a decision function and t correcting functions. However, only the model estimated in the decision ($\mathbf{Z}$) space is used for prediction:

$$f(\mathbf{z}) = (\hat{\mathbf{w}}\cdot\mathbf{z}) + \hat{b} \qquad (2.15)$$

Parameters C and γ control the trade-off between the capacity of decision functions (i.e., margin size), the capacity of correcting functions, and the number of training errors. Setting γ to zero yields standard SVM formulation. Assuming nonlinear kernels for both decision and correcting spaces, SVM+ has 4 tuning parameters (two kernel parameters, C and γ). Model selection with 4 tuning parameters is quite challenging, and resampling approaches with finite data often result in highly variable model estimates.

Correcting functions represent a unique way that SVM+ handles group information. That is, SVM+ approach assumes that the decision function (2.15) interacts

differently with training data from different groups. Since correcting functions model slack variables (in the decision space), they have to be non-negative. That is, samples from each group in the correcting space have to lie on one side of the corresponding correcting function (see Fig. 2.5). Also, a correcting function has to pass through some points with slack variables being zero. The terms $(\mathbf{w}_r \cdot \mathbf{w}_r)$ reflect the capacity of a set of correcting functions. However, this capacity term is not related to the margin size.

Using the dual optimization technique (similar to standard SVM), one can show that $\mathbf{w}$, $\mathbf{w}_r$ in (2.14) can be expressed in terms of training samples:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{z}_i \text{ and } \mathbf{w}_r = \frac{1}{\gamma} \sum_{i \in T_r} (\alpha_i + \beta_i - C) \mathbf{z}_i^r .$$

where α_i and μ_i are found by solving the following dual problem:

$$\begin{aligned} \min_{\alpha,\beta} \quad & -\sum_{i=1}^{n} \alpha_i + \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{z}_i \mathbf{g} \mathbf{z}_j) + \frac{1}{2\gamma} \sum_{r=1}^{t} \sum_{i,j \in T_r} (\alpha_i + \beta_i - C)(\alpha_j + \beta_j - C)(\mathbf{z}_i^r, \mathbf{z}_j^r) \\ \text{subject to: } & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & \sum_{i \in T_r} (\alpha_i + \beta_i) = |T_r| C, \ \alpha_i \geq 0, \ \beta_i \geq 0, \ i \in T_r, r = 1, \text{K}, t \end{aligned} \tag{2.16}$$

2.4.3 LUPI vs. LWSD

The two settings LUPI and LWSD are similar that both settings estimate a single model and additional information is utilized during training. The LWSD can be considered as a special case of LUPI. The group information under LWSD setting can be encoded as a categorical variable $\mathbf{x}^*$, which can be used as privileged information in LUPI. So, the LWSD setting can be converted into LUPI. However, LWSD can provide more flexible

representation of group information. For example, in SVM+ under LWSD, each group r has a unique correcting function $\xi_r(\mathbf{x}_i) = (\mathbf{w}_r \cdot \mathbf{z}_i^r) + d_r \geq 0,$ defined in a unique correcting space Z_r. The coefficient vectors $\mathbf{w}_r$, which control the capacity of correcting functions, are different for different groups. On the other hand, SVM+ under LUPI only has one correcting function: $\xi(\mathbf{x}^*) = \mathbf{w}^* \mathbf{z}^* + d$.

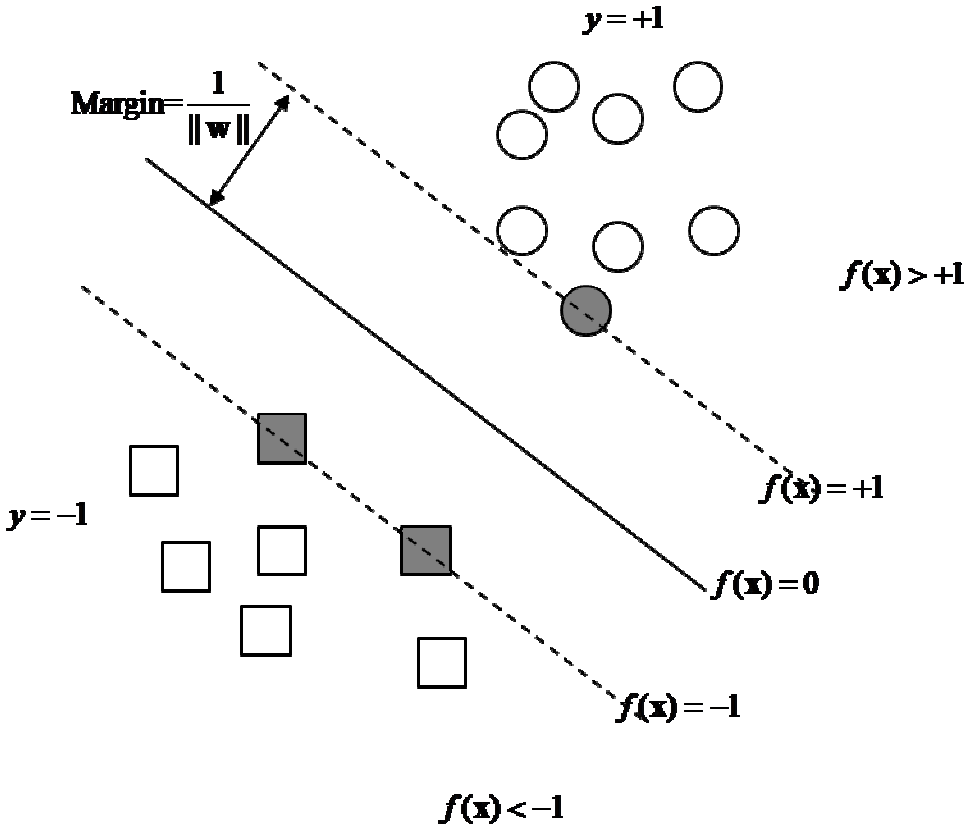

Figure 2.1 Binary classification problem, where ◯ ' denotes samples from positive class and ' □ ' denotes samples from negative class. The margin is the distance between the closest data points to the hyperplance.

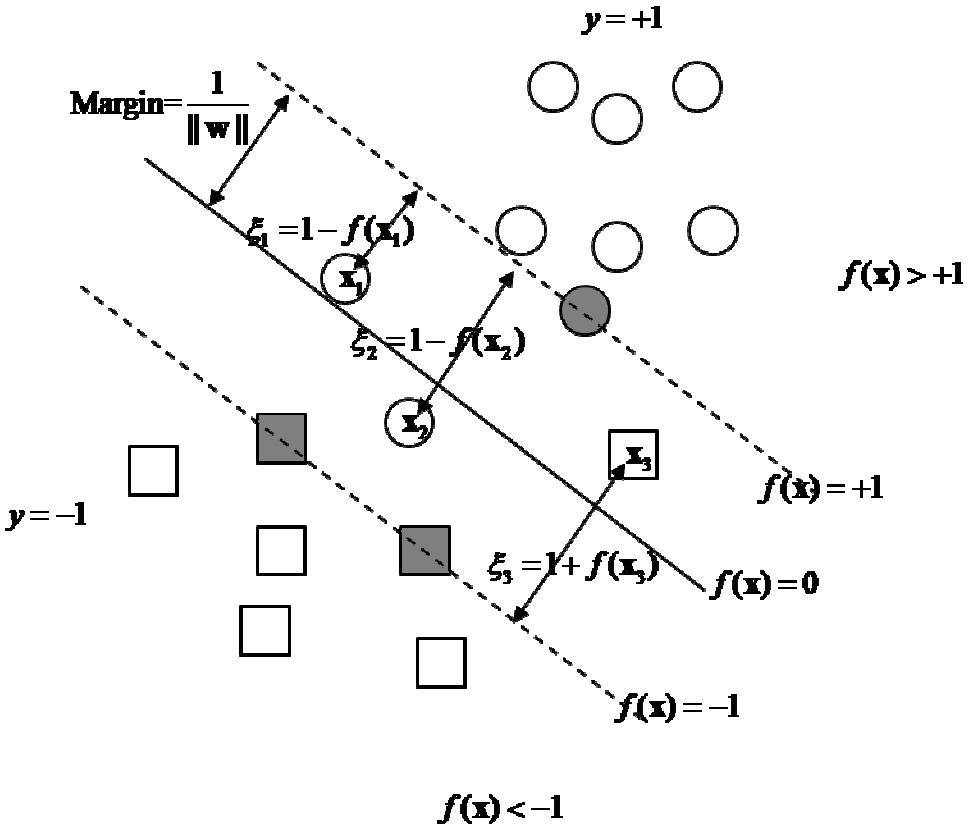

Figure 2.2 Non-separable case for binary classification. Slack variables ξ_i correspond to the deviation from the margin borders.

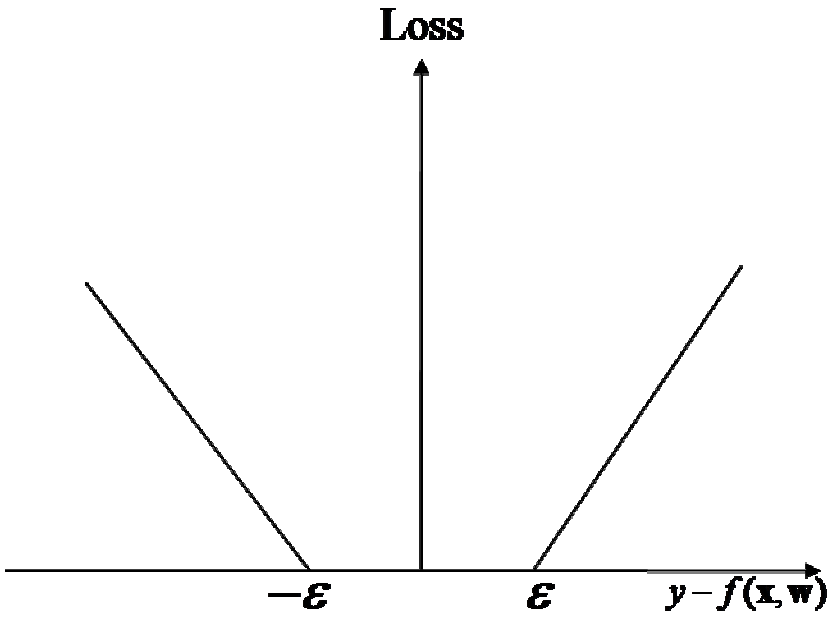

Figure 2.3 ε -insensitive loss function for SVM regression

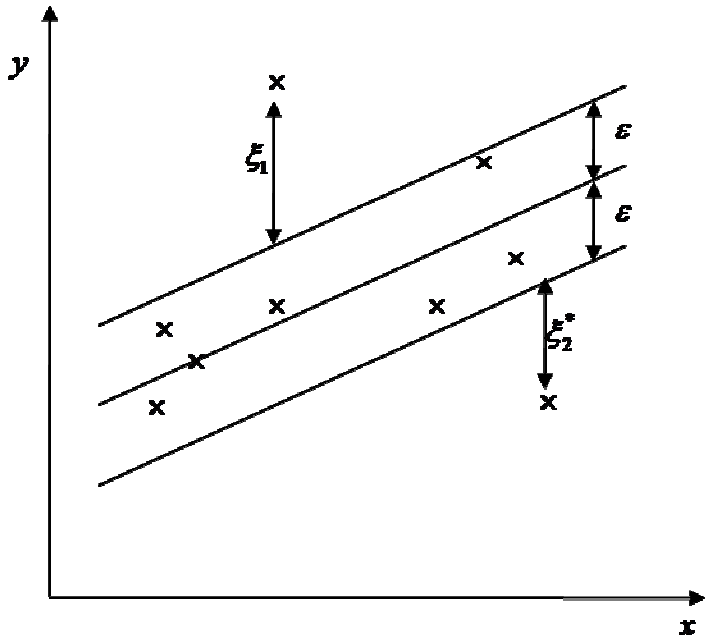

Figure 2.4 slack variable ξ for linear SVM regression formulation

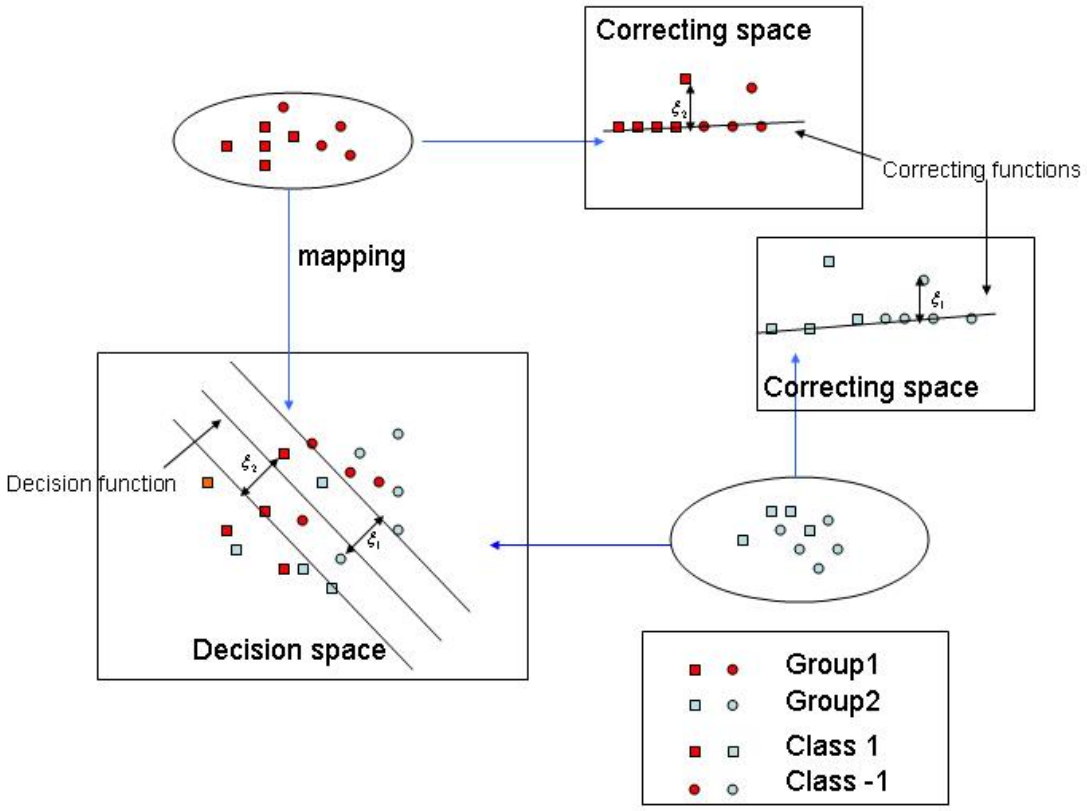

Figure 2.5 SVM+ maps data from two groups simultaneously into decision space and correcting space. Slack variables (in the decision space) are represented by correcting functions defined in the correcting space.

Chapter 3 SVM+ Based Feature Selection

This chapter describes a SVM+ (Vapnik, 1982-2006) based feature selection scheme. Empirical results show that SVM+ can help improve generalization during feature selection.

3.1 Introduction

Estimation of predictive classification models from high dimensional data is becoming increasingly important in various applications such as gene microarray analysis, image based object recognition, functional magnetic resonance imaging (fMRI) analysis etc. Such high-dimensional data sets present challenges for classification learning methods. One way of solving the problem is to use feature selection techniques to reduce the dimensionality of data. There are many potential benefits of feature selection: facilitating data visualization and data understanding, reducing the measurement and storage requirements, reducing training and utilization times, defying the curse of dimensionality to improve prediction performance (Guyon and Elisseeff, 2003). In this chapter, we only focus on improving prediction performance. Feature selection algorithms typically fall into two categories: filter and wrapper (John, Kohavi and Pfleger, 1994). Filter methods select a subset of 'informative' features independently of the predictor (classifier). Wrapper methods incorporate feature selection into the process of learning (estimating) a classifier. In this chapter, we only consider filter methods. The purpose is to investigate how/whether the predictive performance of a given filter method can be improved by

using SVM+. Filter methods work as follows: given training pairs $(\mathbf{x}_i, y_i), i = 1,\text{K}\ , n$, where samples $\mathbf{x}_i$ has d different features: $x_{ij}, j = 1,\text{K}\ , d$, a metric is computed for each feature. Then all the features are ranked by this metric and a predefined number of top-ranked features is selected. A learning system with feature selection is shown in Figure 3.1. In Fig 3.1 (a), feature selection is applied to training data $(\mathbf{x}, y)$ and a subset of features, denoted as input $\mathbf{x}'$, is generated. So the original input $\mathbf{x}$ can be represented as $\mathbf{x} = (\mathbf{x}', \mathbf{x}^*)$ where only selected features $\mathbf{x}'$ are used during training shown in Fig 3.1(b). The learning machine then uses training pairs $(\mathbf{x}', y)$ to estimate a model $f(\mathbf{x}')$. During test in Fig 3.1 (c), for each input $\mathbf{x}'$, the model $f(\mathbf{x}')$ returns predicted output $\hat{y}$ (class label).

Note that feature selection process in Fig 3.1 effectively discards features $\mathbf{x}^*$ which are not used for training. These removed features can be considered as extra information during training (they are not available during test). Recently, Vapnik (1982-2006) proposed SVM+ under Learning Using Privileged Information setting. SVM+ can effectively utilize extra information of training data to help improve generalization. Therefore, in this chapter, we will use SVM+ to help improve generalization after feature selection.

3.2 SVM+ Based Feature Selection

The classical paradigm of supervised machine learning is described as follows: given

finite number of samples called training data $(\mathbf{x}_i, y_i), (i = 1, \text{K}, n)$, the goal of learning is to select the best predictive model $f(\mathbf{x}, \omega^*)$ from a set of functions $f(\mathbf{x}, \omega), \omega \in \Omega$ where Ω is a set abstract parameters used to index the set of functions.

In Chapter 2, we reviewed the SVM+ methodology under LUPI setting, which can utilize additional privileged information $\mathbf{x}^*$ not available in traditional training pairs $(\mathbf{x}, y)$. More formally, under LUPI setting, we are given a set of triplets $(\mathbf{x}_i, \mathbf{x}_i^*, y_i), \mathbf{x}_i \in X, \mathbf{x}_i^* \in X^*, y_i \in \{+1, -1\}, i = 1, \text{K}, n$ generated according to a fixed but unknown probability $p(\mathbf{x}, \mathbf{x}^*, y)$, where $(\mathbf{x}_i, y_i)$ is the 'usual' training data and $(\mathbf{x}_i^*)$ denotes additional *privileged* information. The goal is to find among a given set of functions $f(\mathbf{x}, \omega), \omega \in \Omega$ the optimal function $f(\mathbf{x}, \omega^*)$ which guarantees the smallest probability of incorrect classification. SVM+ estimates the decision function $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{z} + b$ by solving the following problem:

$$\begin{aligned} \min_{\mathbf{w}, b, \mathbf{w}^*, d} \quad & \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + \frac{\gamma}{2}(\mathbf{w}^* \cdot \mathbf{w}^*) + C\sum_{i=1}^{n} \xi_i \\ \text{subject to:} \quad & y_i(\mathbf{w} \cdot \mathbf{z}_i + b) \geq 1 - \xi_i, \ i = 1, \text{K}, n \\ & \xi_i = \mathbf{w}^* \cdot \mathbf{z}^* + d \geq 0, \ i = 1, \text{K}, n \end{aligned} \qquad (3.1)$$

where $C > 0$ and $\gamma > 0$ are hyperparameters, $\mathbf{z}$ and $\mathbf{z}^*$ are feature maps of $\mathbf{x}$ and $\mathbf{x}^*$. The term $\gamma(\mathbf{w}^* \cdot \mathbf{w}^*)/2$ is intended to restrict the capacity (or VC-dimension) of the correcting function.

Now we have already discussed SVM+ under LUPI setting. This new method can utilize additional privileged information to help improve generalization. Next we will show how SVM+ can be incorporated into a learning system with feature selection.

A SVM+ based learning system with feature selection is shown in Figure 3.2. Note that in this figure, the feature selection (a) and test (c) are same as in Fig 3.1. The only difference is Fig 3.2(b), where SVM+ trains a model $f(\mathbf{x}')$ over new training triplets $(\mathbf{x}', \mathbf{x}^*, y)$, where $\mathbf{x}'$ is a set of features generated by feature selection in Fig 3.2(a) and $\mathbf{x}^*$ is vector formed by remaining features. Next we will show that privileged information $\mathbf{x}^*$ helps improve generalization performance using empirical comparisons.

3.3 Empirical Comparisons

The section shows empirically how SVM+ improves generalization using privileged information or features removed by feature selection as in the system shown in Figure 3.2. This section presents comparisons between two methods for classifier training using traditional filter method: standard SVM as in Fig 3.1(b) and SVM+ as in Fig 3.2(b). Linear kernel is used for standard SVM and the decision space of SVM+. The RBF kernel is used for correcting space of SVM+. That is, SVM has only one parameter C and SVM+ has two more parameters: γ and σ (for RBF kernel in correcting space). For all comparisons, we use binary classification setting with equal misclassification costs.

3.3.1 Feature Selection Methods

This section describes several popular filter methods, which rank all the features by a metric and then select a predefined number of top features. Three different feature ranking methods for binary classification are used in this section.

– **Fisher's Criterion** (Fisher, 1936). Suppose we are given training samples $(\mathbf{x}_i, y_i), y_i \in \{-1, +1\}, i = 1, \text{K}, n$ and x_{ij} denotes value of j th input feature of sample i. Let μ_j^+ and σ_j^+ be mean and standard deviation of values of j th feature of positive samples, μ_j^- and σ_j^- be mean and standard deviation of values of j th feature of negative samples, the Fisher's criterion score for j th feature is computed as:

$$F(j) = \frac{(\mu_j^+ - \mu_j^-)^2}{((\sigma_j^+)^2 + (\sigma_j^-)^2)}.$$

– **Information Gain** (Guyon and Elisseeff, 2003). Suppose value of j th feature x_j is categorical variable (as in our examples to be discussed), that is, $x_{ij} \in S_j, i = 1, \text{K}, n$, where S_j is a finite set. The output of our binary classification problem also belongs to a finite set: $y \in \{-1, +1\}$. Therefore, the class prior probabilities $P(y)$, distribution of j th feature $P(x_j)$ and the probabilities of joint observations $P(x_j, y)$ can be estimated from the frequency counts. To this end, the information gain between j th feature and the output can be expressed as:

$$IG(x_j, y) = \sum_{x_j \in S_j} \sum_{y \in \{-1, +1\}} P(x_j, y) \log \frac{P(x_j, y)}{P(x_j)P(y)}. \qquad (3.2)$$

The value of this information gain indicates how informative the corresponding feature is regarding the output.

– **Conditional Mutual Information Maximation (CMIM)** (Fleuret, 2004). We just showed information gain (mutual information) between j th feature and the

output in (3.2). The conditional mutual information between j th feature and the output conditioned on k th feature is:

$$IG(x_j, y / x_k) = \sum_{x_k \in S_k} P(x_k) \sum_{x_j \in S_j} \sum_{y \in \{-1,+1\}} P(x_j, y / x_k) \log \frac{P(x_j, y / x_k)}{P(x_j / x_k) P(y / x_k)}$$
$$= \sum_{x_k \in S_k} \sum_{x_j \in S_j} \sum_{y \in \{-1,+1\}} P(x_j, x_k, y) \log \frac{P(x_k) P(x_j, x_k, y)}{P(x_j, x_k) P(x_k, y)} \quad . \quad (3.3)$$

The probabilities $P(x_j, x_k, y), P(x_k), P(x_j, x_k)$ and $P(x_k, y)$ can be estimated from frequency counts. The conditional mutual information denotes the extra information in j th feature about output when k th feature is given. Feature ranking by CMIM keeps a score vector $\mathbf{g}$ which contains a score for every feature. After the choice of k th feature $v(k)$, the score for remaining features can be updated: $g_i = \min_{l \le k} IG(x_i, y / x_{v(k)})$. The score vector $\mathbf{g}$ is initialized using $IG(x_i, y)$ values provided by (3.2). This is a sequential procedure for feature ranking. It's different from the first two methods which calculate each feature importance independently of other features.

3.3.2 Modeling Results

The first example is the digit recognition problem of classifying images of digits 5 and 8 in the MNIST database (Vapnik and Vashist, 2009). The database describes digits as vectors in 28×28 pixel images and contains 5522 and 5652 images of 5 and 8. Each of the 28×28 pixels (input features) is denoted by a 8-bit integer (integers from 0 to 255). We randomly select 300 samples (150 of '5' and 150 of '8') for training and 300 samples

(150 of '5' and 150 of '8') for validation. The rest of samples are used for test. Experimental procedure for this dataset is as follows: Feature selection (Fisher's criterion and Information Gain) is applied on training data to rank the features. The top 50% of the ranked features are selected as the new input $\mathbf{x}'$ and the remaining 50% features are used to form privileged information $\mathbf{x}^*$, which is used by SVM+. SVM and SVM+ train models on training data using pairs $(\mathbf{x}', y)$ and triplets $(\mathbf{x}', \mathbf{x}^*, y)$ respectively. Validation set is used for model selection. SVM has only one tuning parameter C, which is selected by minimizing validation error. SVM+ has three tuning parameters: C, γ and σ. The parameter C of SVM+ is set to be equal to the optimal C selected for SVM and the remaining two parameters of SVM+ are selected to minimize validation error. Finally, test errors for both SVM and SVM+ are evaluated using test set. The experiment is repeated 8 times and the results using two feature selection methods (Fisher's criterion and Information Gain) are shown in Table 3.1 and 3.2. We can see that SVM+ outperforms SVM for every trial.

We then select 10 top ranked features as new input $\mathbf{x}'$ and the remaining features form $\mathbf{x}^*$. The new results are also shown in Table 3.1 and 3.2, which confirm the previous observations. The prediction errors of SVM and SVM+ with top 10 features are much worse than those results with top 50% features and all features. The reason may be that the top 10 selected features don't contain enough information about the output. There are several common features appearing in the top 10 selected features in 8 different trials. The top common features selected by two feature selection methods (Fisher's Criterion and Information Gain) are marked on digits in Figure 3.3 and 3.4. From Fig 3.3 and 3.4,

we can see that these common features are close to each other and contain a lot of information about digits 5 and 8. Distributions of these common features over rank number 1 to 10 are shown in Fig 3.5 and 3.6. We can see that in the 8 different trials, features '488', '489', '516', '406', '407' and '408' tend to have high rank.

Next we use three datasets from *NIPS-03 feature selection challenge* (Guyon and Gunn, 2003). For each of the three datasets, a separate training set, validation set and test set are provided. The statistics of these datasets are shown in Table 3.3. We use the same experimental procedure and model selection strategy as in the digit recognition example. As several datasets have continuous features, we use Fisher's criterion as our feature selection method here. To evaluate the validation and test performance, balanced error rate (BER) is used. The BER is defined to be the average of the errors on each class: $\text{BER} = 0.5(b/(a+b) + c/(c+d))$, where a, b, c and d represent the number of examples falling into each possible outcome in Table 3.4. We choose different number of selected features and results for the three datasets are shown in table 3.5. The results show that SVM+ is always no worse than SVM and SVM+ beats SVM in several cases. For comparison purposes, we also include SVM modeling using all features. For Dexter and Dorothea datasets, SVM and SVM+ models with feature selection are better than SVM without feature selection when the numbers of selected features are small. For Dextter dataset, SVM+ outperforms SVM when the number of selected features are 100, 1000 and 5000.

The last data set we use is a clinical data set collected from a clinical study performed at Adult Bone and Marrow Transplant Center at the University of Minnesota.

The data set has 473 samples, corresponding to donor-recipient characteristics (features) for patients used in the study. The features of each sample include genetic information (215 SNPs) about donor/recipient pairs, as well as clinical inputs and demographic inputs. The goal is to predict transplant related mortality (binary output) using these features. After preprocessing (i.e. remove samples with many missing values), the resulting data has 301 samples (226 labeled 'alive' and 75 labeled 'dead') each with 136 SNPs. Four clinical and demographic features are selected by researchers in medical school: 'agetx'(recipient age at transplant in years), 'donor' (type of donor), 'cond1' (conditioning regimen prior to transplant) and 'race' (race of recipient). A SVM model was estimated using above four clinical and demographic inputs, as well as 9 SNPs (genetic features) selected by Conditional Information Gain Maximization (Fleuret, 2004). This svm model with unequal misclassification costs was shown to have good predictive performance (Feng et al., 2010). To simplify t1he experiment, we assume equal misclassification costs here. We will show SVM+ beats SVM when 9 SNPs are selected by conditional information gain maximization. The experimental procedure for this dataset is as follows: To evaluate test error, we use five-fold cross validation. That is, the whole dataset is partitioned into 5 subsets. Before partition, the data is sorted by 'age' so that 'age' distribution in each subset is similar. For first fold, the first subset is selected as test set and the remaining four subsets form training set. For model selection, we apply another level of five-fold cross validation within the training set to compute validation error. The validation error is used to tune parameters similar to model selection for digit recognition set. That is, parameter C of SVM is tuned by minimizing validation error.

Parameter C of SVM+ is set to be equal to optimal C of SVM. The other two parameters γ and σ of SVM+ are selected to minimize validation error. We train the model with optimal parameters on training set and compute test error on this test set. The process is repeated for four other folds. The test errors for five different folds are reported in table 3.6. We can see that SVM+ performs better than SVM for all folds and that using feature selection is always better than SVM modeling with all features.

The experiments show that SVM+ can utilizes features removed (these features are not available during test) by feature selection and results in improved generalization performance. However, the improvements depend on the characteristics of data and the usefulness of the privileged information.

Table 3.1 Test error (%) for digit recognition data (Fisher's criterion)

Trials	1	2	3	4	5	6	7	8	Mean(std.dev)
SVM(top 50% features)	6.7	5.68	5.9	7.02	7.02	6.22	6.06	6.11	6.37 (0.51)
SVM+(top50% features)	6.53	5.09	5.41	6.38	6.75	6.10	5.84	5.30	6.01 (0.61)
SVM (top 10 features)	13.29	15.59	13.61	12.92	12.54	12.49	13.24	13.4	13.39 (0.98)
SVM+(top 10 features)	13.07	13.77	12.43	12.43	12.11	12.00	11.89	13.12	12.60 (0.66)
SVM(All features)	6.43	5.94	6.11	6.54	7.02	6.11	6.53	6.43	6.39（0.34）

Table 3.2 Test error (%) for digit recognition data (Information Gain)

Trials	1	2	3	4	5	6	7	8	Mean(std.dev)
SVM(top 50% features)	6.00	5.04	6.32	7.23	6.54	5.84	5.73	5.63	6.1（0.66）
SVM+(top50% features)	5.94	4.66	5.89	6.69	6.16	5.78	5.57	5.51	5.81 (0.58)
SVM (top 10 features)	11.67	13.83	12.92	13.4	14.15	10.5	12.97	14.09	12.78 (1.27)
SVM+(top 10 features)	11.36	12.16	12.37	13.29	12.05	10.18	12.86	13.71	12.25 (1.12)
SVM(All features)	6.16	5.09	6.48	7.02	6.05	5.94	5.73	5.52	6.00 (0.59)

Table 3.3 Statistics of three datasets from NIPS 2003 feature selection challenge

Name	Type	Num. ex. (training/validation/test)	Num. ex. Training (pos/neg)	Num. Feat.
Arcene	Non-sparse	100/100/700	44/56	10000
Dexter	Sparse-integer	300/300/2000	150/150	20000
Dorothea	Sparse-binary	800/350/800	78/722	100000

Table 3.4 Confusion Matrix

		Prediction	
		Class -1	Class +1
Truth	Class -1	A	B
	Class +1	C	D

Table 3.5 Test errors % for NIPS challenge datasets

Num of Features	SVM	SVM+	SVM with all features
100	33.97	33.97	15.12
1000	23.33	23.20	15.12
2000	22.66	22.82	15.12
4000	18.83	17.11	15.12

(a) Arcene dataset

Num of Features	SVM	SVM+	SVM with all features
100	8.55	7.55	10.25
1000	10.20	8.15	10.25
5000	9.05	7.65	10.25
10000	10.25	10.25	10.25

(b) Dexter dataset

Num of Features	SVM	SVM+	SVM with all features
100	26.61	26.61	39.38
1000	26.54	26.54	39.38
10000	38.60	38.60	39.38
50000	41.74	41.74	39.38

(c) Dorothea dataset

Table 3.6 Test error (%) for medical data

Folds	1	2	3	4	5	Mean(Std. dev)
SVM with feature selection	26.23	26.67	23.33	26.67	23.33	25.25 (1.76)
SVM+ with feature selection	22.95	25	18.33	23.33	21.67	22.26 (2.50)
SVM without feature selection	26.23	33.33	25	28.33	23.33	27.25 (3.86)
Majority Voting	27.87	26.67	20	26.67	23.33	24.91 (3.22)

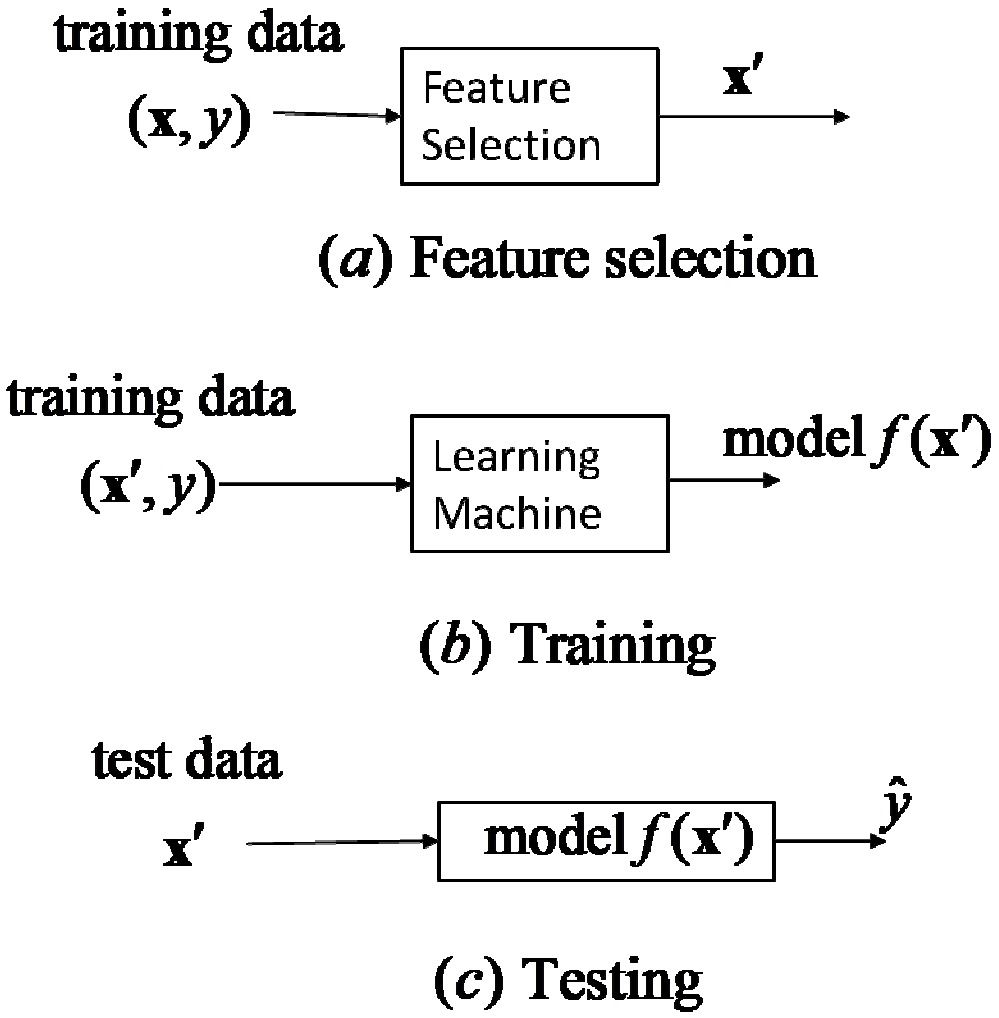

Figure 3.1 Traditional learning system using filter feature selection. In (a), feature selection is applied to training data $(\mathbf{x}, y)$ **and a low dimensional input** $\mathbf{x}'$ **is generated. In (b), learning machine trains a model** $f(\mathbf{x}')$ **over new training pairs** $(\mathbf{x}', y)$**. During test in (c), for given input** $\mathbf{x}'$ **, the model** $f(\mathbf{x}')$ **returns an output** $\hat{y}$**.**

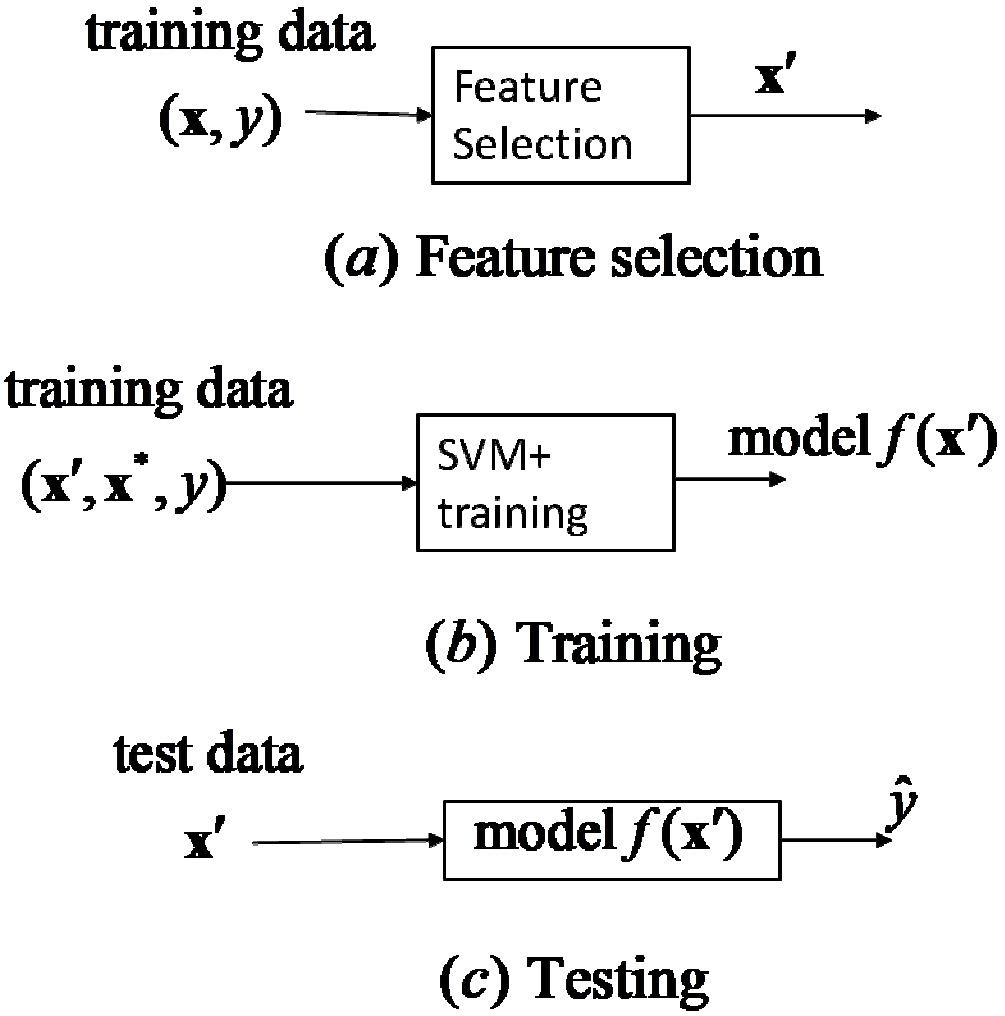

Figure 3.2 A SVM+ based learning system using filter feature selection. In (a), feature selection is applied to training data $(\mathbf{x}, y)$ and a low dimensional input $\mathbf{x}'$ is generated. In (b), SVM+ trains a model $f(\mathbf{x}')$ over new training triplets $(\mathbf{x}', \mathbf{x}^*, y)$, where $\mathbf{x}'$ is a set of features generated by feature selection in (a) and $\mathbf{x}^*$ is vector formed by remaining features . During test in (c), for given input $\mathbf{x}'$, the model $f(\mathbf{x}')$ returns an output $\hat{y}$. Note that the feature selection (a) and test (c) are same as in Fig 3.1.

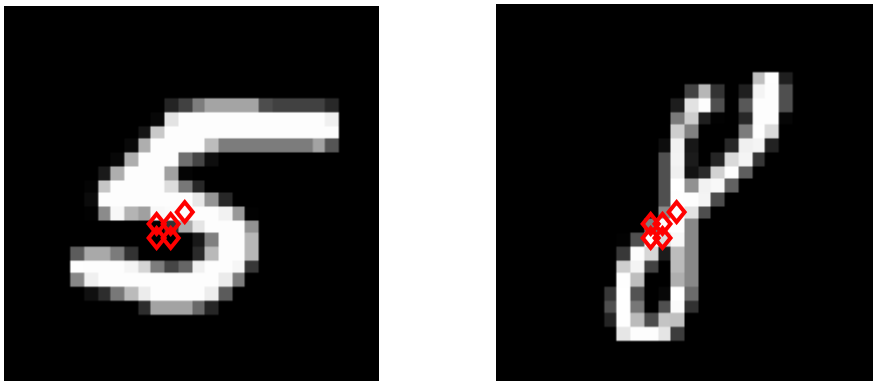

Figure 3.3 Five common features selected by Fisher's criterion are marked on digits 5 and 8. These five features appear in top 10 features selected by Fisher's criterion in 8 different trials.

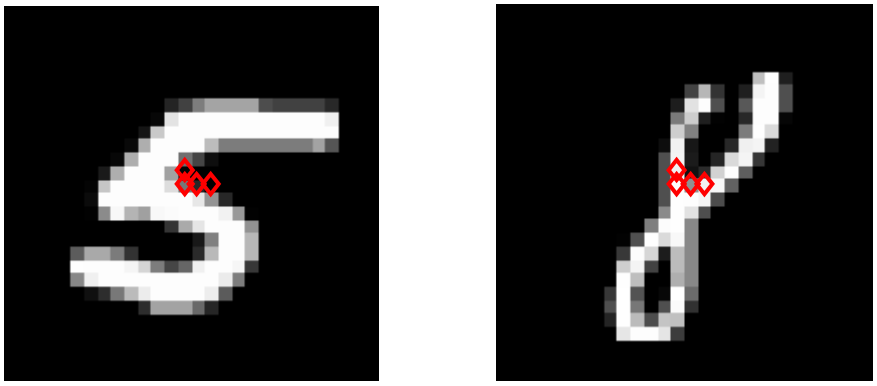

Figure 3.4 Four common features selected by Information Gain are marked on digits 5 and 8. These five features appear in top 10 features selected by Information Gain in 8 different trials.

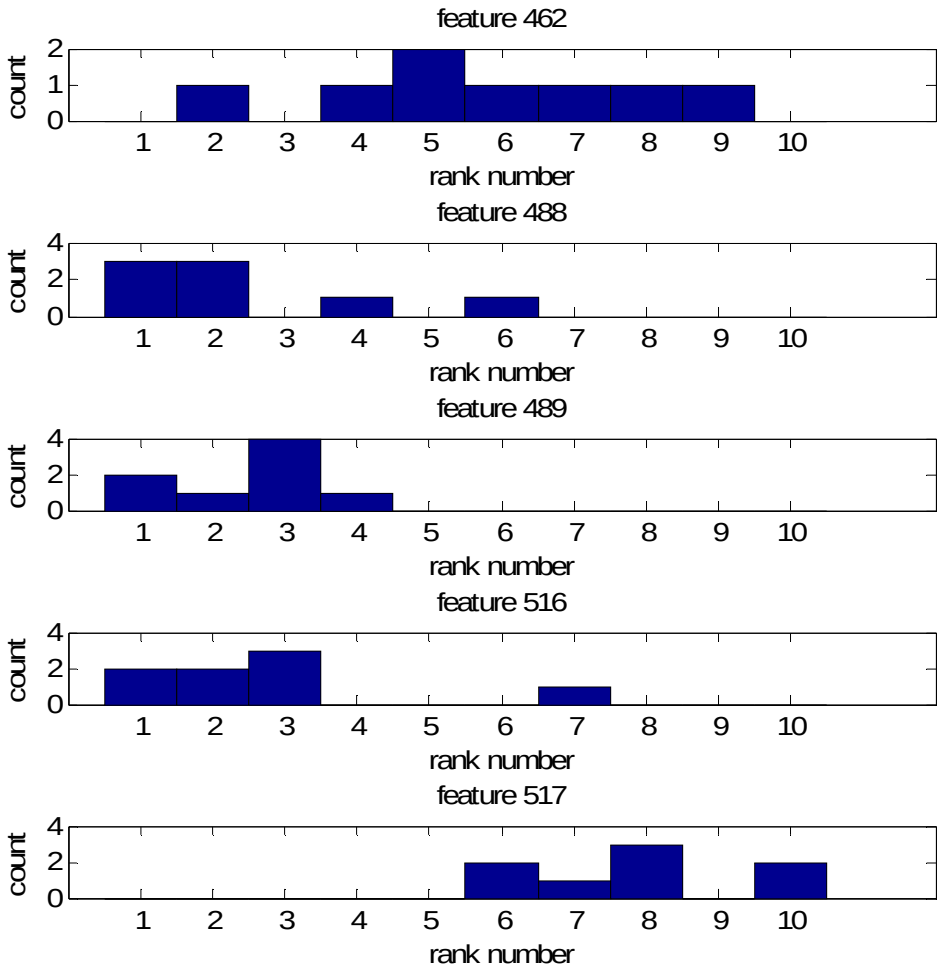

Figure 3.5 Distribution of common features appearing in top 10 features ranked by Fisher's Criterion in 8 different trials. From the distribution, we can see that feature '488', '489' and '516' tend to have high rank.

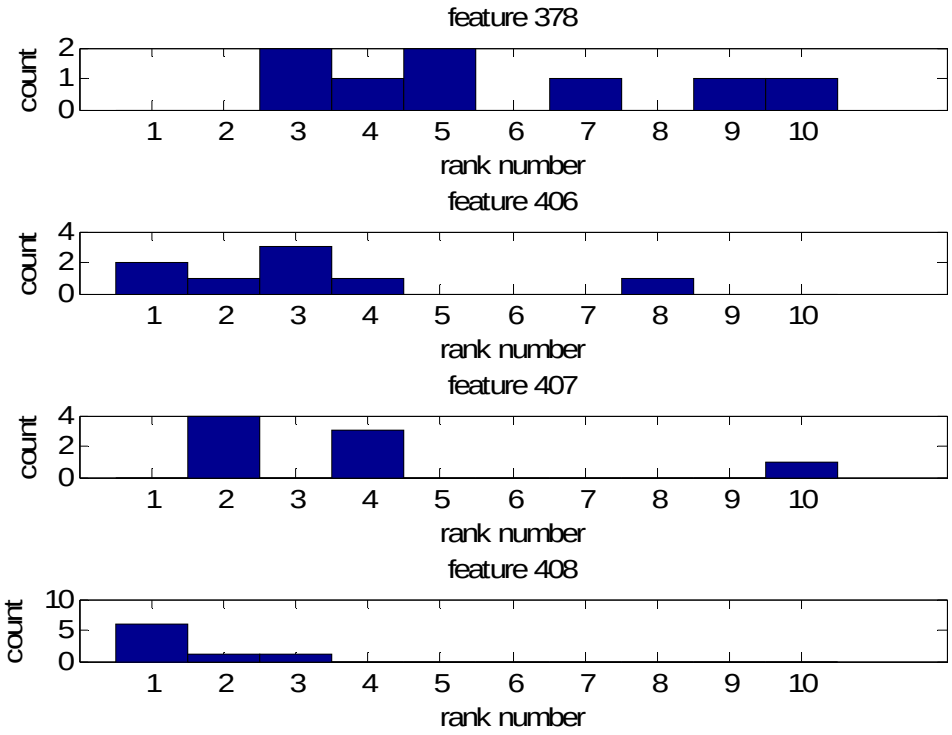

Figure 3.6 Distribution of common features appearing in top 10 features ranked by Information Gain in 8 different trials. From the distribution, we can see that feature '406', '407' and '408' tend to have high rank.

Chapter 4 SVM+ Based Multi-Task Learning for Classification

This chapter studies new methods under classification settings where training data are naturally separated into several related groups. Effectively utilizing this group information during training often results in improved generalization. Several methods such as SVM+ (Vapnik, 2006), regularized MTL (rMTL) (Evgeniou and Pontil, 2004) and SVM -Based Multi-Task Learning (SVM+MTL) (Liang and Cherkassky, 2008) are reviewed and compared. We also incorporate unequal misclassification cost into SVM+MTL classifiers.

4.1 Introduction

There is a growing need for development of powerful and robust methods for estimating predictive models from data. Most supervised learning methods developed in statistics, pattern recognition, and neural networks assume standard inductive learning (aka single task learning (STL)) setting (Vapnik, 1998; Cherkassky and Mulier, 2007; Hastie, T., 2001), where we are given a training set $\mathbf{D} = \{(\mathbf{x}_i, y_i), (i = 1, \mathrm{K}, n)\}$ with samples identically and independently generated from an unknown probability distribution $p(\mathbf{x}, y)$. The goal is to find the best mapping function $f : \mathbf{x} \to y$ such that the expected loss

$$R_{STL}(\omega) = \iint_{X,Y} L(f(\mathbf{x}, \omega), y)\, p(\mathbf{x}, y) d\mathbf{x} dy \qquad (4.1)$$

is minimized. Here in (1.1) $L(f(\mathbf{x}, \omega), y)$ is used to denote a loss function. It can be 0/1 classification error for classification problems.

In this chapter, we consider supervised learning setting where the training data includes additional group information. Examples include: (1) handwritten digit recognition where training samples are provided by several persons, (2) medical diagnosis where predictive (diagnostic) model, say for lung cancer, is estimated using a training data set of male and female patients, etc. Incorporating this additional group information has lead to two different settings known as Learning With Structured Data (LWSD) (Vapnik, 2006) and Multi-Task Learning (MTL).

Formally, suppose the training data comes from t related groups. For each group r, we have n_r data points sampled from a distribution p_r on $\mathbf{X} \times \mathbf{Y}$. So the data is a union of $t > 1$ related groups: $\{\{X_r, Y_r\}, r = 1, \mathrm{K}, t\}, \{X_r, Y_r\} = \{\{\mathbf{x}_{r_1}, y_{r_1}\}, \mathrm{K}, \{\mathbf{x}_{r_{n_r}}, y_{r_{n_r}}\}\}$ and can be thought as samples identically and independently generated from the distribution $p = \cup_{r=1,\mathrm{K},t} p_r$.

If the group labels of future test samples are not given, as in *handwritten digit recognition* example, the setting is LWSD and the goal is to find one best mapping function f such that the expected loss

$$R_{LWSD}(\omega) = \iint_{X,Y} L(f(\mathbf{x}, \omega), y)\, p(\mathbf{x}, y) d\mathbf{x} dy \qquad (4.2)$$

is minimized. The difference between (4.1) and (4.2) is that P in (4.1) is an unknown probability while P in (4.2) is known to be a union of t sub-distributions. Recently, a SVM based method under LWSD setting, known as SVM+ (to be discussed later), was proposed (Vapnik, 2006).

On the other hand, if the group labels of future test samples are given, as in *medical diagnosis* example, the setting is MTL and the goal is to find t mapping functions $\{f_1, f_2, \text{K}, f_t\}$ such that the sum of expected loss for each task

$$R_{MTL}(\omega) = \sum_{r=1}^{t} \iint_{X,Y} L(f_r(\mathbf{x}, \omega), y)\, p_r(\mathbf{x}, y) d\mathbf{x} dy \qquad (4.3)$$

is minimized.

Most MTL technologies can be broadly grouped into several categories: sharing internal representation (Bakker and Heskes, 2003; Ando and Zhang, 2005; Liao and Carin, 2005), learning priors over parameters (Raina, Ng and Koller, 2006; Lawrence and Platt, 2004), learning relevant shared features via joint sparse regularization (Argyriou, Evgeniou and Pontil, 2006; Obozinski et al., 2006; Amit et al., 2007), and kernel methods and regularization (Evgeniou and Pontil, 2004; Liang and Cherkassky, 2008; Cai and Cherkassky, 2009). SVM+MTL and rMTL, which will be discussed in the thesis, falls into the last category.

4.2 SVM+ Approach (Vapnik, 2006)

We already reviewed SVM+ technology in chapter 2. Formally, assume that labeled training data can be represented as a union of t disjoint subsets (or groups), $\{T_r\}$ $r \in [1,2,..,t]$, where each group r contains n_r samples. In order to be useful for learning, SVM+ relates this group information to training errors, or slack variables ξ_i used in the standard soft-margin SVM formulation (see chapter 2). The group information is used to introduce additional constraints on the slack variables (errors) for samples from different

groups by correcting functions $\xi_r(\mathbf{x}_i) = (\mathbf{w}_r \cdot \mathbf{z}_i^r) + d_r \geq 0,\ r = \{1,...,t\}$. These correcting functions $\xi_r(\mathbf{x}_i)$ for group T_r, are defined in the correcting space $\mathbf{Z}_r$. That is, the input vectors $\mathbf{x}_i, i \in T_r$ are mapped into two different spaces:

- the decision space $\mathbf{Z}$ using mapping $\Phi(\mathbf{x}): \mathbf{x} \to \mathbf{Z}$ (as in standard SVM), and
- the correcting space $\mathbf{Z}_r$ via mapping $\Phi_{Z_r}(\mathbf{x}): \mathbf{x} \to \mathbf{Z}_r$. The correcting functions for different groups are specified in $\mathbf{Z}_r$ space.

To this end, SVM+ estimates the single decision function $f(\mathbf{z}) = (\hat{\mathbf{w}} \cdot \mathbf{z}) + \hat{b}$ by solving the following SVM+ optimization formulation:

$$\begin{aligned} \min_{\substack{\mathbf{w},b,\mathbf{w}_r,d_r \\ r=1,\mathrm{K},t}} \quad & \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + \frac{\gamma}{2}\sum_{r=1}^{t}(\mathbf{w}_r \cdot \mathbf{w}_r) + C\sum_{r=1}^{t}\sum_{i \in T_r}\xi_i^r \\ \text{subject to:} \quad & y_i((\mathbf{w} \cdot \mathbf{z}_i) + b) \geq 1 - \xi_i^r, \qquad (4.4) \\ & \xi_i^r = (\mathbf{w}_r \cdot \mathbf{z}_i) + d_r \geq 0,\ i \in T_r,\ r = 1,\mathrm{K},t. \end{aligned}$$

Parameters C and γ control the trade-off between the capacity of decision functions (i.e., margin size), the capacity of correcting functions, and the number of training errors. Setting γ to zero yields standard SVM formulation. Assuming nonlinear kernels for both decision and correcting spaces, SVM+ has 4 tuning parameters (two kernel parameters, C and γ). Model selection with 4 tuning parameters is quite challenging, and resampling approaches with finite data often result in highly variable model estimates.

Correcting functions represent a unique way that SVM+ handles group information. That is, SVM+ approach assumes that the decision function interacts differently with training data from different groups. Since correcting functions model slack variables (in the decision space), they have to be non-negative. The terms $(\mathbf{w}_r \cdot \mathbf{w}_r)$

reflect the capacity of a set of correcting functions. However, this capacity term is not related to the margin size.

Using the dual optimization technique (similar to standard SVM), one can show that $\mathbf{w}$, $\mathbf{w}_r$ in (4.4) can be expressed in terms of training samples:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{z}_i \text{ and } \mathbf{w}_r = \frac{1}{\gamma} \sum_{i \in T_r} (\alpha_i + \beta_i - C) \mathbf{z}_i^r .$$

where α_i and μ_i are found by solving the following dual problem:

$$\min_{\alpha,\beta} \quad -\sum_{i=1}^{n} \alpha_i + \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{z}_i \mathbf{z}_j) + \frac{1}{2\gamma} \sum_{r=1}^{t} \sum_{i,j \in T_r} (\alpha_i + \beta_i - C)(\alpha_j + \beta_j - C)(\mathbf{z}_i^r, \mathbf{z}_j^r)$$

$$\text{subject to: } \sum_{i=1}^{n} \alpha_i y_i = 0 \qquad (4.5)$$

$$\sum_{i \in T_r} (\alpha_i + \beta_i) = |T_r| C, \ \alpha_i \geq 0, \ \beta_i \geq 0, \ i \in T_r, r = 1, \text{K}, t$$

Therefore, the decision function has new form:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i (\mathbf{z}_i \mathbf{z}) + b \qquad (4.6)$$

4.3 SVM+MTL Approach (Liang and Cherkassky, 2008)

MTL setting is similar to LWSD, in the sense that available training data is naturally segmented into several groups, corresponding to different related classifiers (or 'tasks'). However, the difference is that for MTL the group (task) label is also known for test data. So under MTL the goal is to estimate several related classifiers, whereas SVM+ approach estimates a single classifier.

Now we describe an SVM-based approach to multi-task learning (Liang and Cherkassky, 2008), where different tasks share a common part in their decision functions. So the goal of learning is to estimate t classifiers, or t decision functions, in the form:

$$f_r(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{z}) + b + \phi_r(\mathbf{x}), \;\; r = \{1,...,t\} \qquad (4.7)$$

where $\mathbf{z} = \Phi(\mathbf{x})$ denotes kernel mapping. Here the decision function for each classifier, or task, $f_r(\mathbf{x})$ includes two parts: a common term $(\mathbf{w} \cdot \Phi(\mathbf{x})) + b$, and a unique term $\phi_r(\mathbf{x})$ for task r. The common part is estimated in a decision space obtained via a kernel mapping $\Phi(\mathbf{x}) : \mathbf{x} \to \mathbf{Z}$, as in standard SVM. There may be different ways to parameterize the unique term $\phi_r(\mathbf{x})$. One approach, motivated by SVM+, is to define these unique terms in a different 'correcting' space $\mathbf{Z}_r$. That is, the unique terms for task r is specified in the correcting space $\mathbf{Z}_r$ as $\varphi_r(\mathbf{z}^r, \mathbf{w}_r)$ where $\mathbf{z}^r = \Phi_{Z_r}(\mathbf{x})$. To this end, SVM+MTL tries to estimate t related classifiers in the form:

$$f_r(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{z}) + b + (\mathbf{w}_r \cdot \mathbf{z}^r) + d_r, \;\; r = \{1,...,t\} \qquad (4.8)$$

where $\mathbf{z} = \Phi(\mathbf{x})$ and $\mathbf{z}^r = \Phi_{Z_r}(\mathbf{x})$. Decision functions (4.8) are estimated from the training data. Training data for MTL setting can be described using the notation used earlier in Section 3.1. That is, labeled training data is represented as a union of t disjoint subsets (for different tasks), $\{T_r\}_{\; r \in [1,2,..,t]}$, where each group r contains n_r samples. Then the input vectors $\mathbf{x}_i, i \in T_r$ are mapped into two spaces: the decision space $\mathbf{Z}$ using mapping $\Phi(\mathbf{x}) : \mathbf{x} \to \mathbf{Z}$ and the correcting space $\mathbf{Z}_r$ via mapping $\Phi_{Z_r}(\mathbf{x}) : \mathbf{x} \to \mathbf{Z}_r$. The unique terms for each task r are estimated in the correcting space.

However, the optimization formulation for SVM+ MTL is different from SVM+, because under multi-task learning the goal is to estimate t related classifiers. Decision functions (4.8) are estimated from the training data for all tasks $(\mathbf{x}_i, y_i), i \in T_r$, $r \in [1,2,..,t]$ as a solution to the following optimization problem:

$$\begin{aligned} \min_{\substack{\mathbf{w},b,\mathbf{w}_r,d_r \\ r=1,\mathrm{K},t}} \quad & \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + \frac{\gamma}{2}\sum_{r=1}^{t}(\mathbf{w}_r\cdot\mathbf{w}_r) + C\sum_{r=1}^{t}\sum_{i\in T_r}\xi_i^r \\ \text{subject to:} \quad & y_i((\mathbf{w}\cdot\mathbf{z}_i) + b + (\mathbf{w}_r\cdot\mathbf{z}_i^r) + d_r) \geq 1 - \xi_i^r, \\ & \xi_i^r \geq 0,\ i \in T_r,\ r = 1,\mathrm{K},t. \end{aligned} \qquad (4.9)$$

Here, the norms of $\mathbf{w}$ and $\mathbf{w}_r$ are used to control the capacity of the common decision function and the correcting function, respectively. Parameter γ adjusts the relative weight of these two capacities, and C controls the trade-off between complexity and number of non-separable samples. The slack variables ξ_i^r measure the error that each task model makes on the data. Here each task model includes the common part and task-specific correcting function, according to (4.8).

This formulation (4.9) has some similarity to the regularized multi-task learning (rMTL) proposed by Evgeniou and Pontil (2004), which also assumes each decision function has a common part and a unique part:

$$f_r(\mathbf{x}) = (\mathbf{w}_0 + \mathbf{w}_r)\cdot\mathbf{z},\ r = 1,\mathrm{K},t. \qquad (4.10)$$

where $\mathbf{z} = \Phi(\mathbf{x})$ denotes kernel mapping. Coefficient vectors $\mathbf{w}_0$ and $\mathbf{w}_r$, $r = 1,\mathrm{K},t$ can be found by solving the following optimization problem.

$$\min_{\substack{\mathbf{w}_0, b, \mathbf{w}_r, d_r \\ r=1,\mathrm{K},t}} \quad \frac{1}{2}(\mathbf{w}_0 \cdot \mathbf{w}_0) + \frac{\gamma}{2}\sum_{r=1}^{t}(\mathbf{w}_r \cdot \mathbf{w}_r) + C\sum_{r=1}^{t}\sum_{i\in T_r}\xi_i^r$$

$$\text{subject to:} \quad y_i(\mathbf{w}_0 + \mathbf{w}_r)\cdot \mathbf{z}_i \geq 1 - \xi_i^r, \qquad (4.11)$$

$$\xi_i^r \geq 0,\ i \in T_r,\ r = 1,\mathrm{K},t.$$

The main difference between rMTL and SVM+MTL is that under rMTL formulation inputs map to one space, whereas under SVM+MTL formulation inputs map to two different spaces: decision space and correcting space.

The dual form of (4.9) is as follows:

$$\min_{\alpha} \quad -\sum_{i=1}^{n}\alpha_i + \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j (\mathbf{z}_i \cdot \mathbf{z}_j) + \frac{1}{2\gamma}\sum_{r=1}^{t}\sum_{i,j\in T_r}\alpha_i\alpha_j y_i y_j (\mathbf{z}_i^r \cdot \mathbf{z}_j^r)$$

$$\text{subject to:} \quad \sum_{i\in T_r}\alpha_i y_i = 0,\ 0 \leq \alpha_i \leq C,\ i \in T_r, r = 1,\mathrm{K},t. \qquad (4.12)$$

Based on Karush-Kuhn-Tucker (KKT) conditions, we can express $\mathbf{w}$ and $\mathbf{w}_r$ of (4.9) in terms of training samples:

$$\mathbf{w} = \sum_{i=1}^{n}\alpha_i y_i \mathbf{z}_i \ \text{ and } \ \mathbf{w}_r = \frac{1}{\gamma}\sum_{i\in T_r}\alpha_i y_i \mathbf{z}_i^r$$

$$\text{Thus,} \quad f_r(\mathbf{x}) = \sum_{i=1}^{n}\alpha_i y_i (\mathbf{z}_i \cdot \mathbf{z}) + b + \frac{1}{\gamma}\sum_{i\in T_r}\alpha_i y_i (\mathbf{z}_i^r \cdot \mathbf{z}^r),\ r = 1,\mathrm{K},t. \qquad (4.13)$$

4.4 SVM+MTL with Unequal Cost

In previous sections, we assume that misclassification costs are same for different classes. However, in many real applications, misclassification costs are class dependent. For example, in medical diagnosis, the cost of patient with heart disease misclassified as absence of heart disease and the cost of patient without heart disease misclassified as presence of heart disease are different (Michie et al., 1994). Actually, unequal

misclassification costs were already incorporated into standard SVM (Vapnik, 1998; Cherkassky and Mulier, 2007). Formally, assume we have a binary classification problem: $y_i \in \{+1, -1\}$. Let C_{fp} denotes cost of negative class misclassified as positive class and C_{fn} cost of positive class misclassified as negative class, the SVM with unequal costs solves the following problem (Vapnik, 1998; Cherkassky and Mulier, 2007):

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C_{\text{fn}} \sum_{y_i=+1} \xi_i + C_{\text{fp}} \sum_{y_i=-1} \xi_i \\ \text{subject to:} \quad & y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i, \ i = 1, \ldots, n. \end{aligned} \tag{4.14}$$

Similary, we propose to incorporate unequal misclassification costs into formulation of SVM+MTL (4.9):

$$\begin{aligned} \min_{\substack{\mathbf{w},b,\mathbf{w}_r,d_r \\ r=1,\ldots,t}} \quad & \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + \frac{\gamma}{2}\sum_{r=1}^{t}(\mathbf{w}_r\cdot\mathbf{w}_r) + C_{\text{fn}}\sum_{r=1}^{t}\sum_{i\in T_r, y_i=+1}\xi_i^r + C_{\text{fp}}\sum_{r=1}^{t}\sum_{i\in T_r, y_i=-1}\xi_i^r \\ \text{subject to:} \quad & y_i((\mathbf{w}\cdot\mathbf{z}_i) + b + (\mathbf{w}_r\cdot\mathbf{z}_i^r) + d_r) \geq 1 - \xi_i^r, \\ & \xi_i^r \geq 0, \ i \in T_r, \ r = 1, \ldots, t. \end{aligned} \tag{4.15}$$

To make this formulation closer to the original formulation (4.9), we define a vector **s** as follows:

$$s_i = \begin{cases} 1 & \text{if } y_i = 1, \\ C_{\text{fp}} / C_{\text{fn}} & \text{if } y_i = -1. \end{cases}$$

Then, formulation (4.15) can be changed to:

$$\begin{aligned} \min_{\substack{\mathbf{w},b,\mathbf{w}_r,d_r \\ r=1,\ldots,t}} \quad & \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + \frac{\gamma}{2}\sum_{r=1}^{t}(\mathbf{w}_r\cdot\mathbf{w}_r) + C\sum_{r=1}^{t}\sum_{i\in T_r} s_i\xi_i^r \\ \text{subject to:} \quad & y_i((\mathbf{w}\cdot\mathbf{z}_i) + b + (\mathbf{w}_r\cdot\mathbf{z}_i^r) + d_r) \geq 1 - \xi_i^r, \\ & \xi_i^r \geq 0, \ i \in T_r, \ r = 1, \ldots, t. \end{aligned} \tag{4.16}$$

Similar to standard SVM, the primal problem (4.16) is solved by solving its dual problem:

$$\max_{\alpha} \quad \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j (z_i, z_j) - \frac{1}{2\gamma}\sum_{r=1}^{t}\sum_{i,j\in T_r}\alpha_i\alpha_j y_i y_j (z_i^r, z_j^r)$$

$$\text{s. t.} \quad \sum_{i\in T_r}\alpha_i y_i = 0,\ r = 1,\text{K}\ ,t$$

$$0 \le \alpha_i \le s_i C,\ i = 1,\text{K}\ ,n. \qquad (4.17)$$

Comparing the dual problem of standard SVM+MTL (4.12) and the dual problem of SVM+MTL with unequal cost (4.17), we find that the only difference is the inequality constraint. That is, $0 \le \alpha_i \le C$ in (4.12) is slightly changed to $0 \le \alpha_i \le s_i C$ in (4.17). Therefore, the training algorithm for original SVM+MTL can be slightly modified to train the SVM+MTL with unequal costs.

4.5 Empirical Comparisons

This section describes empirical comparisons to validate SVM+MTL and SVM+MTL with unequal costs.

4.5.1 SVM+MTL vs. Single SVM, Multiple SVMs and SVM+

Assuming that available training data can be partitioned (in a meaningful way) into several groups, we can identify several learning approaches for utilizing this group information. Several possibilities for incorporating the group (task) information into SVM-style modeling are shown in Figure 4.1. They include:

- sSVM ~ single (or standard) SVM classifier that pools together training data from several tasks and uses this combined data set for training standard SVM;

- SVM+ ~ available information about group (task) labels in the training data is used to estimate a single SVM+ model as described in Section 4.2;
- mSVM ~ multiple SVM classifiers, estimated *independently* for each task, or each subset of the training data;
- SVM+MTL ~ modeling approach implementing multi-task learning, which estimates several related classification models simultaneously as described in Section 4.3.

In Fig. 4.1, the class labels are represented by circles and squares, and the group membership is denoted by filled (dark) and non-filled symbols. Empirical comparisons of these learning approaches for synthetic data can be found in (Liang and Cherkassky, 2008; Liang 2008). Next we will present empirical comparisons of these learning approaches for real data.

Now we consider a real data set corresponding to radar-based sensing of landmines (Xue et al., 2007). The dataset has 29 tasks, where each task corresponds to data collected from a different landmine field. Each data sample is represented by a 9-dimensional feature vector and the corresponding binary label (1 for landmine and 0 for clutter). Data from 6 different tasks (tasks 1, 2, 11, 20, 22, 24) are selected and number of samples in each task is varied from 40 to 160. For each task, one quarter of training samples correspond to landmines, and the rest to clutters. For estimating the prediction error of a learning method, we use 5-fold cross-validation such that every 5^{th} sample in each group is used as test data, and the remaining samples constitute training data. For each training fold, parameter tuning (model selection) is performed via another level of 5-fold cross

validation. We use stratified resampling (Kitagawa, 1996) so that the fraction of positive/negative class samples in test data is the same as in training data. We use RBF kernels for the decision space in all SVM methods, and use RBF kernels for decision space and for correcting space in SVM+ and SVM+MTL. Table 4.1 shows the prediction error for several methods. For this data, single model methods SVM and SVM+ don't perform well and the performance becomes worse as the size in each group varies from 40 to 160. Multiple model methods mSVM and SVM+MTL perform much better. Over all methods, SVM+MTL is the best. This is a typical dataset where data comes from different tasks/groups. Data from different groups may have very different characteristics, which is why the single model method SVM and SVM+ don't perform well. SVM+MTL performs better than mSVM because SVM+MTL trains multiple related models simultaneously.

Next we compare the four methods on several publicly available medical data sets from UCI machine learning repository. At first glance, these datasets are not collected from different groups. But we can manually select a group variable from a list of input variables and partition the available data into several groups (tasks) corresponding to different values (or range of values) of the group variable. Each group is roughly of similar size. The sSVM uses all features and all other three methods use all features but the group variable. Comparisons use both linear and RBF kernels for sSVM and mSVM. Common decision space for SVM+ and SVM+MTL uses linear kernel while the unique correction space uses RBF kernel. To estimate prediction error, we use double resampling.

Comparison results show an average test error (averaged over 5 folds) and its standard deviation.

A. Statlog heart disease dataset

There are 270 instances, each of which has 13 attributes. The goal is to predict an absence or presence of heart disease using 13 input variables. Variable 'SEX' is used to separate the data into two groups: group1($SEX = 0$, 87 instances)and group2($SEX = 1$, 183 instances). Comparison results are shown in Table 4.2 (a).

B. Ljubljana breast cancer dataset

It has 286 instances, each with 9 attributes. The dataset contains 9 instances with missing values, so only the remaining 277 instances are used. The goal is to predict the class (no-recurrence-events or recurrence-events) from 9 attributes. Variable *'age'* was selected to separate the data into 3 different groups: group 1($age < 47$, 94 instances), group 2($47 \le age < 55$, 93 instances) and group 3($age \ge 55$, 90 instances). Results are shown in Table 4.2 (b).

C. Wisconsin breast cancer dataset

There are 699 instances, each of which has 9 continuous attributes. The measurements of attributes are assigned an integer value between 1 and 10. After removing 16 instances with missing values, we are left with 683 instances used for modeling. The goal is to predict the class (*benign* or *malignant*) using 9 input variables. Variable 'Clump Thickness' is used to separate the data into 3 groups: group1 (Clump Thickness < 4, 293 instances), group2($4 \le$ Clump Thickness<6, 207 instances),

and group3 (Clump Thickness ≥ 6 , 183 instances). Comparison results are shown in Table 4.2 (c).

D. Hepatitis dataset

There are 155 instances, each of which has 19 attributes. After removing 75 instances with missing values, we are left with 80 instances for modeling. The goal is to predict the binary class label (*dead* or *alive*) using 19 input variables. We separate data into 2 groups using binary variable 'HISTOLOGY': group 1($HISTOLOGY = 1$, 47 instances) and group 2($HISTOLOGY = 2$, 33 instances). Results are shown in Table 4.2 (d).

All modeling results shown above suggest an improvement in generalization performance due to partitioning of the data into several informative groups. However, selection of 'good' group variable for partitioning the data is still based on intuition and application domain knowledge. So next we discuss the problem of effective specification of a group variable and its effect on generalization performance. Specifically, we compare selection of different group variables for the Ljubljana breast cancer dataset. For this dataset, goal is to predict the class (non-recurrence or recurrence of breast cancer) from 9 attributes: *age, menopause, tumor-size, inv-nodes, node-caps, degree-of-malignancy, breast, breast-quad, irradiat*. The absolute value of Pearson correlation coefficients between each attribute variable and the output class variable are shown in Table 4.3. The large coefficients indicate strong correlation with the output.

Experimental results in Table 4.4 show what happens when selected group variable has strong/weak correlation with the output. When a group variable is strongly correlated

with the output (i.e., *deg-malig*, *inv-nodes*), different groups tend to be quite independent of each other. In this case, multiple model methods (mSVM, SVM+MTL) tend to perform better (no worse) than single model methods (sSVM, SVM+), as evident from Table 4.4 (a) and (b). In particular, simple mSVM approach that estimates several independent SVM models shows quite good performance. The more complex SVM+MTL method does not provide an improvement over mSVM, since there is little information shared among groups. Also, the performance of SVM+MTL may have been degraded due to unbalanced sample size in different groups.

On the other hand, if a group variable (i.e. *age*) has small correlation coefficient, there may be considerable overlap among different groups, or interdependency between tasks. More information can be shared among groups/tasks, and this leads to improved performance of multi-task approaches SVM+ and SVM+MTL (see Table 4.4 (c)). However, small correlation coefficient may be also due to irrelevant (noisy) input. For example, the input variable *breast* (with two values, left or right) has very small correlation coefficient, and this input is clearly irrelevant for prediction. So this group variable (breast) cannot divide the data into meaningful groups. Hence, advanced multi-task learning approach SVM+MTL does not yield much improvement over mSVM (as shown in Table 4.4 (d)).

4.5.2 SVM+MTL vs. rMTL

This section describes empirical comparisons between SVM+MTL and rMTL. Since the main difference between these two methods is that rMTL projects inputs into one space whereas SVM+MTL projects inputs into two different spaces: decision space and

correcting space, it is expected that SVM+MTL performs better than rMTL. Linear kernel is used in decision space for SVM+MTL while RBF kernel is used for rMTL and in correcting space for SVM+MTL. We use same synthetic data as in (Liang and Cherkassky, 2008) and same real data sets as in Section 4.5.1. For these two methods, we follow the same model selection and experimental procedure for SVM+MTL as in Section 4.5.1. The synthetic data is generated as follows:

(1) Let number of input features be d=20, and number of tasks (groups) be t=3.

(2) Generate $\mathbf{x} \in R^{20}$ with each element $x_i \sim uniform(-1,1)$, $i = 1,...,20$.

(3) For each task, the output class label is specified as $y = sign\left((\beta, \mathbf{x}) + 0.5\right)$.

where the coefficient vectors for each task are given as

$\beta_1 = [1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0]$
$\beta_2 = [1,1,1,1,1,1,1,1,1,0,1,0,1,0,0,0,0,0,0,0]$
$\beta_3 = [1,1,1,1,1,1,1,1,0,1,0,0,0,0,0,1,0,0,0,0]$

Note that the decision boundaries for 3 tasks have a common part, i.e. first seven coefficients of vectors β are the same.

Table 4.5 shows the test results of two methods on synthetic data. We change the sample size per task from 100 to 15 and the results are averaged over 10 different trials. The results show that SVM+MTL performs better than rMTL in all cases. The results for several medical data sets are shown in Table 4.6. We observe that SVM+MTL outperforms rMTL in the first three data sets in terms of the average prediction errors. The rMTL only achieves a slightly better prediction error in *Wisconsin breast cancer dataset*.

From the experimental results, we observe that SVM+MTL is better in most cases. As we stated earlier, the advantage of SVM+MTL is that it projects inputs into two different spaces: decision space and correcting space, whereas rMTL only projects inputs into one space. This gives SVM+MTL more flexibility to fit the training data from different groups. However, SVM+MTL has one more tuning parameter corresponding to kernel of the extra space. This makes model selection for SVM+MTL more complex and difficult than rMTL.

4.5.3 SVM+MTL with Unequal Cost vs. SVM with Unequal Cost and Standard SVM+MTL

This section empirically shows the advantage of SVM+MTL with Unequal Cost, where unequal misclassification costs are incorporated into training of SVM+MTL. We compare SVM+MTL with unequal costs with two other methods: SVM with unequal costs and SVM+MTL with equal costs. To evaluate the predictive performance, we use the following weighted error (Michie et al., 1994):

$$\text{Error}_{\text{weighted}} = \frac{N_{\text{fn}} + N_{\text{fp}} \times C_{\text{fp}} / C_{\text{fn}}}{N_{\text{p}} + N_{\text{n}} \times C_{\text{fp}} / C_{\text{fn}}} \qquad (4.18)$$

where C_{fp} and C_{fn} denote cost of false positive and false negative errors respectively, N_{p}, N_{n}, N_{fp} and N_{fn} denote numbers of positive samples, negative samples, false positive samples and false negative samples respectively. A synthetic data and a medical data are used in this section.

- **Synthetic data**

The synthetic data set similar to the data in (Liang and Cherkassky, 2008) is generated as follows:

(1) Let number of input features be d=30, and number of tasks (groups) be t=3.

(2) Generate $\mathbf{x} \in R^{30}$ with each element $x_i \sim uniform(-1,1),\ \ i = 1,...,30$.

(3) For each task, the output class label is specified as $y = sign\left(\left(\beta \mathbf{x}\right) + 4\right)$.
where the coefficient vectors for each task are given as

$$\beta_1 = [1,1,2,3,3,1,1,1,1,0,2,0,2,2,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0]$$
$$\beta_2 = [1,1,2,3,3,1,1,1,0,2,0,2,2,0,0,0,0,2,0,2,0,0,0,0,0,0,0,0,0]$$
$$\beta_3 = [1,1,2,3,3,1,1,0,1,0,0,3,0,0,2,0,2,0,2,0,0,0,0,0,0,0,0,0,0]$$

For each task, we generate 200 data samples for training, 600 samples for validation and 600 samples for test. The ratio of costs is specified as: $C_{\text{fp}} / C_{\text{fn}} = 5$. Table 4.7 shows the weighted prediction errors for the three different methods. We observe that SVM+MTL with unequal costs performs much better than the other two methods. It is straightforward that SVM+MTL with unequal costs outperforms SVM with unequal costs since SVM+MTL with unequal costs effectively utilizes the group information. The other observation that SVM+MTL with unequal costs performs better than SVM+MTL without unequal costs is also expected. We will show next that the incorporation of unequal costs into SVM+MTL training helps move the decision boundary towards class of samples where misclassification cost is smaller.

To illustrate the differences of two SVM+MTL models graphically, we reduce both feature dimension and number of groups to 2, and the modified data is generated as follows:

(1) Let number of input features be $d = 2$, and number of tasks (groups) be $t = 2$.

(2) Generate $\mathbf{x} \in R^2$ with each element $x_i \sim uniform(-1,1),\ \ i = 1,2..$

(3) For each task, the output class label is specified as $y = sign\left((\beta \mathbf{x}) + 0.3\right)$.
where the coefficient vectors for each task are given as

$$\beta_1 = [1,\ 0.2]$$
$$\beta_2 = [1,\ 0.4]$$

Similary, for each task, we generate 30 samples for training and 600 samples for validation. The ratio of costs is specified as: $C_{\text{fp}} / C_{\text{fn}} = 3/2$. Figure 4.2 shows decision boundaries for two different SVM+MTL models. We observe that decision boundaries of two classifiers estimated by SVM+MTL with unequal costs moved slightly right toward the positive samples. This explains the smaller weighted prediction error attained by SVM+MTL with unequal costs as false negative cost is much smaller in this experiment.

- **Statlog heart data set**

This medical data set has 270 samples, each with 13 features and one binary output ('1' for absence and '2' for presence of heart disease). The label '1' is encoded as '-1' and '2' is encoded as '+1'. One output 'Age' is used to partition the data into two groups: male and female. The cost matrix of this data, specified by a statlog study (Michie et al., 1994), is shown in Table 4.8. Therefore, the ratio of misclassification costs is: $C_{\text{fp}} / C_{\text{fn}} = 0.2$. A double resampling technique was used to evaluate the model performance, i.e. resampling for estimating true weighted prediction error, and resampling for model selection (parameter tuning). The experimental setup is as follows:

(1) Perform resampling for estimating true weighted prediction error, by taking out samples 1, 10, 19,… as test data, and the remaining samples as training data. This will form one fold of 9-fold cross validation, by splitting all the data into 8+1 parts; then do the same with samples 2, 11, 20, … (second fold) etc.

(2) Within each training fold (above), perform an additional 9-fold cross validation on the data ordered by age, in the same manner as described above, for model selection.

Table 4.9 shows the weighted prediction errors for three different methods. This confirms the observation from synthetic data experiment that SVM+MTL with unequal costs improves generalization.

4.6 Conclusion

This chapter described several classification methods which can effectively utilize group information in training data. We experimentally validated SVM+ under LWSD setting (Vapnik, 2006), where group information is utilized to train a single model. We then described a MTL method extended from SVM+: SVM+MTL (Liang and Cherkassky, 2008). Empirical comparisons showed that SVM+MTL is better than standard SVM and rMTL (another MTL method, Evgeniou and Pontil, 2004). Finally, we proposed to incorporate unequal misclassification costs into SVM+MTL formulation. Empirical results showed that the incorporation of unequal costs help improve generalization of SVM+MTL.

Table 4.1 Test errors % for landmine dataset

Size in each group	40	80	160
sSVM (test error %)	38.33 ± 5.81	45.00 ± 2.38	53.02 ± 8.06
mSVM (test error %)	18.33 ± 7.28	11.46 ± 3.68	14.90 ± 6.76
SVM+ (test error %)	35.00 ± 6.74	35.42 ± 5.16	51.35 ± 5.28
SVM+MTL (test error %)	15.00 ± 5.19	10.00 ± 2.72	9.69 ± 2.35

Table 4.2 Test errors for medical datasets

Method	sSVM(linear)	sSVM(rbf)	SVM+
Test error %	19.3 ± 7.5	18.2 ± 6.5	16.3 ± 6.1
Method	mSVM	mSVM(rbf)	SVM+MTL
Test error %	16.6 ± 4.3	21.5 ± 5.3	15.2 ± 4.0

(a) Statlog heart dataset

Method	sSVM(linear)	sSVM(rbf)	SVM+
Test error %	29.3 ± 6.2	25.7 ± 4.5	24.9 ± 4.8
Method	mSVM(linear)	mSVM(rbf)	SVM+MTL
Test error %	29.6 ± 1.6	24.2 ± 2.5	23.5 ± 3.4

(b) Ljubljana breast cancer dataset

Method	sSVM(linear)	sSVM(rbf)	SVM+
Test error %	3.4 ± 1.3	3.8 ± 0.8	3.1 ± 1.0
Method	mSVM(linear)	mSVM(rbf)	SVM+MTL
Test error %	3.4 ± 0.8	3.1 ± 1.0	2.9 ± 0.9

(c) Wisconsin breast cancer dataset

Method	sSVM(linear)	sSVM(rbf)	SVM+
Test error %	16.3 ± 8.4	17.5 ± 5.2	16.3 ± 8.4
Method	mSVM(linear)	mSVM(rbf)	SVM+MTL
Test error %	16.3 ± 8.4	16.3 ± 8.4	15.0 ± 7.1

(d) Hepatitis dataset

Table 4.3 Absolute value of Pearson correlation coefficient between attributes and the output (Ljubljana breast cancer dataset)

Attributes	age	Menopause	tumor-size	inv-nodes	node-caps	deg-malig	Breast	breast-quad	Irradiat
Coefficient	0.0675	0.0588	0.1784	0.3027	0.0289	0.2994	0.0586	0.0910	0.1939

Table 4.4 Test error with different group variables (Ljubljana breast cancer dataset)

Method	sSVM(linear)	sSVM(rbf)	SVM+	mSVM(linear)	mSVM(rbf)	SVM+MTL
Test error %	29.61 ± 6.6	27.07 ± 3.8	27.79 ± 7.7	24.53 ± 5.7	26.34 ± 2.6	24.90 ± 4.7

(a) Group variable *deg-malig*

Method	sSVM(linear)	sSVM(rbf)	SVM+	mSVM(linear)	mSVM(rbf)	SVM+MTL
Test error %	29.63 ± 4.7	26.20 ± 5.2	25.62 ± 5.5	26.70 ± 3.3	25.28 ± 5.0	25.61 ± 4.9

(b) Group variable *inv-nodes*

Method	sSVM(linear)	sSVM(rbf)	SVM+	mSVM(linear)	mSVM(rbf)	SVM+MTL
Test error %	29.3 ± 6.2	25.7 ± 4.5	24.9 ± 4.8	29.6 ± 1.6	24.2 ± 2.5	23.5 ± 3.4

(c) Group variable *age*

Method	sSVM(linear)	sSVM(rbf)	SVM+	mSVM(linear)	mSVM(rbf)	SVM+MTL
Test error %	30.33 ± 4.7	24.57 ± 5.8	27.07 ± 5.3	26.73 ± 5.8	26.02 ± 4.4	26.03 ± 5.4

(d) Group variable *breast*

Table 4.5 The test errors in percentage for synthetic data. Results are averaged over 10 trials (SVM+MTL vs. rMTL).

Method	Training size	Mean (st. dev)
SVM+MTL	100 training samples per task	8.86±1.29
rMTL	100 training samples per task	13.52±1.22
SVM+MTL	50 training samples per task	12.70±1.39
rMTL	50 training samples per task	18.99±1.13
SVM+MTL	15 training samples per task	21.15±3.39
rMTL	15 training samples per task	31.68±3.17

Table 4.6 The test error in percentage for medical data sets (SVM+MTL vs. rMTL)

Data set	Size	Method	1	2	3	4	5	Mean
Statlog	270	SVM+MTL	16.7	18.5	18.5	20.4	11.1	17.0±3.6
		RMTL	24.1	22.2	20.4	18.5	11.1	19.3±5.0
Hepatitis	80	SVM+MTL	25.0	12.5	12.5	18.8	6.3	15.0±7.1
		RMTL	25.0	12.5	12.5	25.0	12.5	17.5±6.8
Ljubljana	277	SVM+MTL	25.5	19.6	25.0	20.0	27.3	23.5±3.4
		RMTL	30.9	21.4	23.2	25.5	29.1	26.0±4.0
Wisconsin	683	SVM+MTL	2.2	2.2	2.9	2.9	4.4	2.9±0.9
		RMTL	1.5	2.2	2.9	2.9	4.4	2.8±1.1

Table 4.7 Weighted test error for synthetic data (Averaged over 10 trials)

Methods	Mean (St. dev)
SVM with Unequal Costs	0.286(0.025)
SVM+MTL with equal costs	0.346(0.032)
SVM+MTL with unequal costs	0.179(0.014)

Table 4.8 Misclassification costs for the statlog heart disease dataset. The columns represent the predicted class, and the rows the true class.

	Absence (‘-1’)	Presence (‘+1’)
Absence (‘-1’)	0	1
Presence (‘+1’)	5	0

Table 4.9 Weighted test errors % for three SVM based methods (Statlog heart data, 9-fold cross validation)

Methods	Mean (st. dev)
SVM with unequal cost	23 (7.7)
SVM+MTL with equal cost	18 (11.7)
SVM+MTL with unequal cost	16 (8.3)

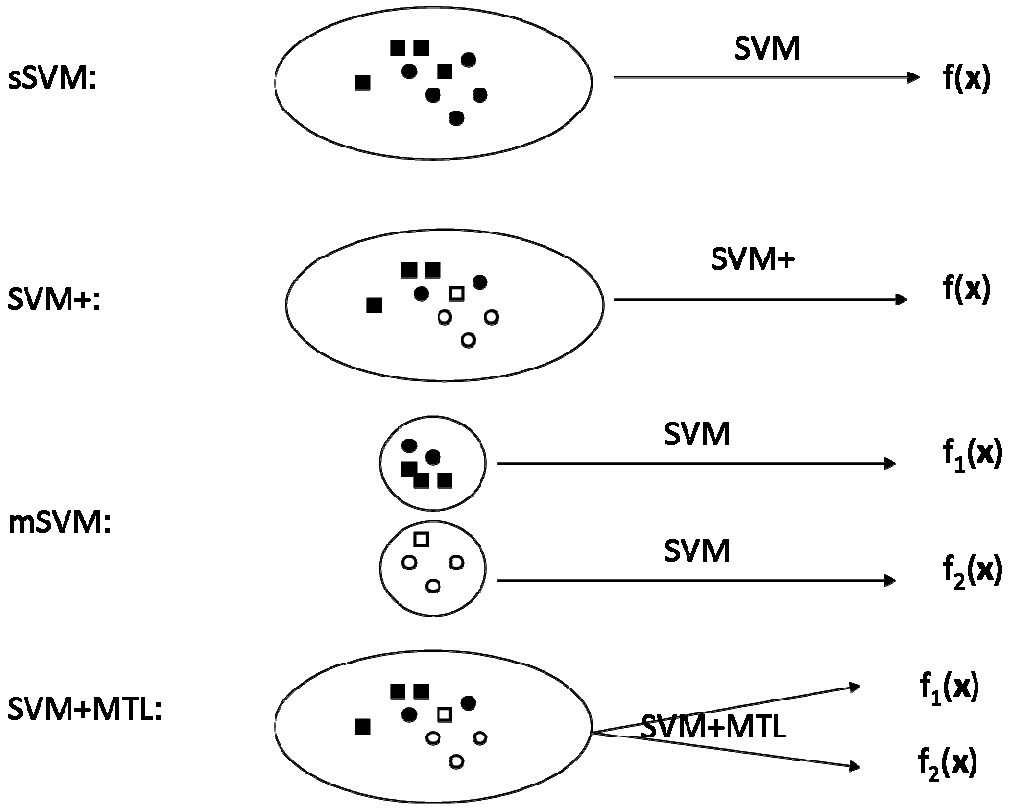

Figure 4.1 Different ways of using group information in learning: (a) sSVM ~ Single SVM classifier, (b) SVM+ classifier, (c) mSVM ~ multiple independent SVMs, and (d) SVM+MTL ~ SVM+ Multi-Task Learning

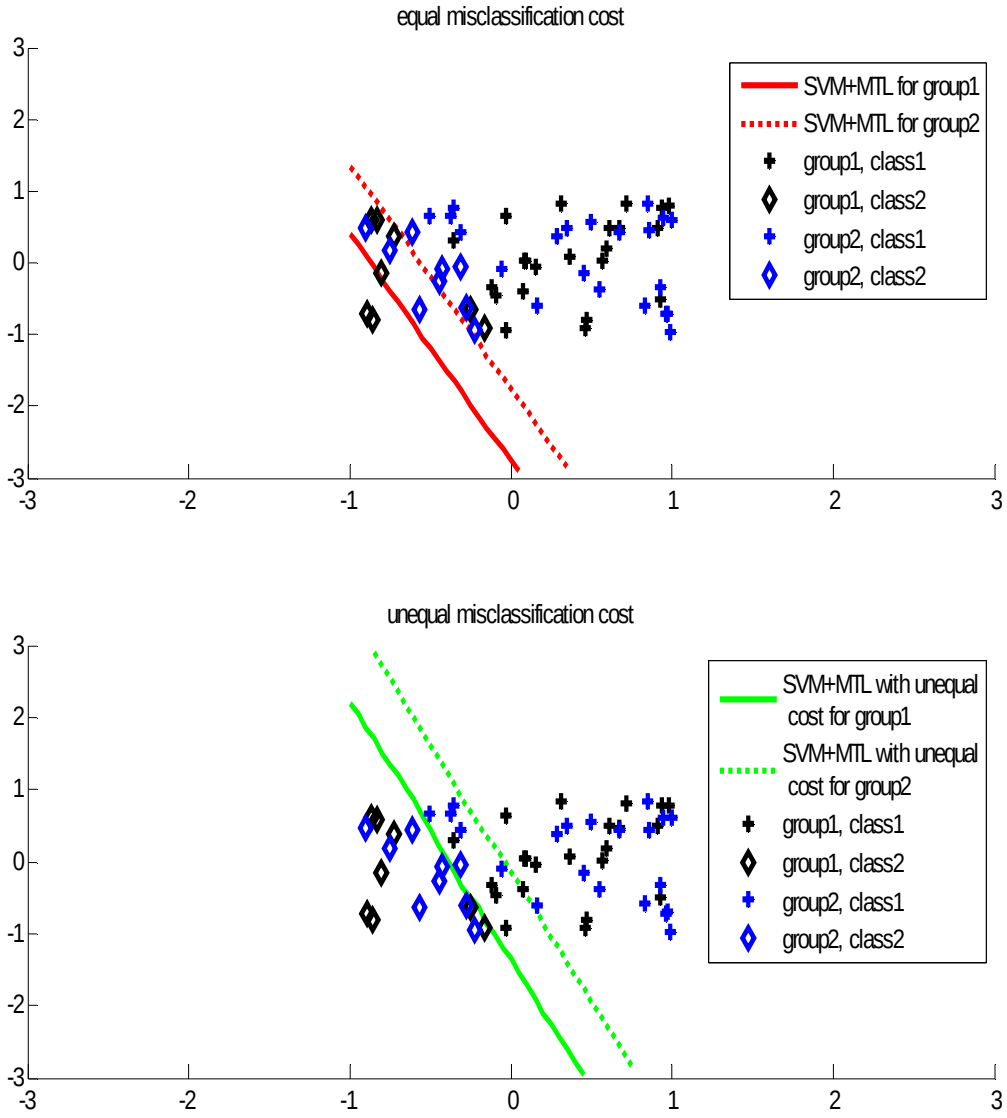

Figure 4.2 Decision boundaries of standard SVM+MTL and SVM+MTL with unequal cost on synthetic data. The positive symbols denote samples in positive class while the diamond symbols denote samples in negative class. The ratio of costs is specified as: $C_{\text{fp}} / C_{\text{fn}} = 3/2$.

Chapter 5 SVM+ Based Multi-Task Learning for Regression

In this chapter, we present new methodology called SVM+MTL regression, combining SVM+ regression (Vapnik, 2006) and SVM+MTL classification (Liang and Cherkassky, 2008). We also show empirical comparisons between standard SVM regression, SVM+ regression and proposed SVM+MTL regression. Practical implementation of new learning technologies, such as SVM+, is often hindered by their complexity, i.e. large number of tuning parameters (vs standard inductive SVM regression). To this end, we provide a practical scheme for model selection that combines analytic selection of parameters for SVM regression (Cherkassky and Mulier, 2007) and resampling-based methods for selecting model parameters specific to SVM+ and SVM+MTL.

5.1 Introduction

In Chapter 3, we discussed several new non-inductive classification methodologies such as SVM+ under LWSD setting and SVM+MTL under MTL setting, where training data can be naturally partitioned into several related groups. Empirical results showed that effective use of group information during training (i.e. in SVM+ and SVM+MTL) can help improve generalization. Similarly, these new non-inductive methods can be applied to regression problems, where training data can be naturally partitioned into several groups. In fact, several MTL methods have already been applied to these regression problems and achieved better prediction accuracy (Evgeniou and Pontil, 2004; Ando and Zhang, 2005; Liao and Carin, 2005). Therefore, in this chapter, we present new

methodology SVM+MTL regression, combining SVM+ regression and SVM+MTL classification.

5.2 SVM+ Regression

SVM+ regression is a non-inductive methodology under LWSD setting, where group information within training data is used to help train a single model. Suppose the training data are the union of $t > 1$ related groups. Let us denote the indices from group r by $T_r = \{i_{r_1}, \text{K}, i_{r_{n_r}}\}, r = 1, \text{K}, t$. Then all training samples can be represented as: $\{\{X_r, Y_r\}, r = 1, \text{K}, t\}, \{X_r, Y_r\} = \{\{\mathbf{x}_{r_1}, y_{r_1}\}, \text{K}, \{\mathbf{x}_{r_{n_r}}, y_{r_{n_r}}\}\}, \mathbf{x} \in R^d, y \in R$. Similar to classification version, SVM+ regression maps vectors in each group $\mathbf{x}_i, i \in T_r$ simultaneously *into two different Hilbert spaces* Z ($\mathbf{z}_i = \phi_z(\mathbf{x}_i) \in Z$) and Z_r ($\mathbf{z}_i^r = \phi_{z_r}(\mathbf{x}_i) \in Z_r$). To account for the group information, Vapnik (2006) defines the slack variables as follows:

$$\xi_i = (\mathbf{w}_r \cdot \mathbf{z}_i) + d_r, i \in T_r, r = 1, ..., t$$

$$\xi_i^{r*} = (\mathbf{w}_r^* \cdot \mathbf{z}_i^r) + d_r^*, i \in T_r, r = 1, ..., t$$

Compared to standard SVM regression, here the slack variables are restricted by the correcting functions, and the correcting functions represent additional information about the data. The goal is to find the regression function in decision space Z:

$$f(\mathbf{x}) = (\mathbf{w} \cdot \phi_Z(\mathbf{x})) + b \qquad (5.1)$$

Note that data of different groups are mapped into the same decision space, and they are all used to construct the regression function. However, there are different correcting functions for different groups. Correcting functions are defined in the

correcting space. Different correcting functions can be defined either in the same correcting space or different correcting function spaces. Of course, if data of different groups are mapped to different correcting spaces, the correcting functions for different groups are different. If data of different groups are mapped to the same correcting space, we still can construct different correcting functions for different groups. The importance is that correcting functions are different, not correcting space.

In the case of two groups, mapping of the training data by SVM+ regression is schematically shown in Figure 5.1.

Formally, SVM+ regression approach estimates regression model $f(\mathbf{x}) = (\mathbf{w} \cdot \phi_z(\mathbf{x})) + b$ by solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w},\mathbf{w}_r,\mathbf{w}_r^*,b,\mathbf{d}_r,\mathbf{d}_r^*} \quad & \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + \frac{\gamma}{2}\sum_{r=1}^{t}((\mathbf{w}_r \cdot \mathbf{w}_r) + (\mathbf{w}_r^* \cdot \mathbf{w}_r^*)) + C\sum_{r=1}^{t}\sum_{i \in T_r}(\xi_i^r + \xi_i^{r*}) \\ \text{subject to:} \quad & y_i - (\mathbf{w} \cdot \mathbf{z}_i) - b \le \varepsilon + \xi_i^{r*},\ \xi_i^{r*} \ge 0 \\ & (\mathbf{w} \cdot \mathbf{z}_i) + b - y_i \le \varepsilon + \xi_i^r,\ \xi_i^r \ge 0 \\ & \xi_i^r = (\mathbf{w}_r \cdot \mathbf{z}_i^r) + d_r,\ \xi_i^{r*} = (\mathbf{w}_r^* \cdot \mathbf{z}_i^{r*}) + d_r^*,\ i \in T_r,\ r = 1,\text{K},t. \end{aligned} \tag{5.2}$$

Using the dual optimization technique, one can show that $\mathbf{w}$, $\mathbf{w}_r$ and $\mathbf{w}_r^*$ can be expressed in terms of training samples:

$$\mathbf{w} = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)\mathbf{z}_i ,$$

$$\mathbf{w}_r = \frac{1}{\gamma}\sum_{i \in T_r}(\alpha_i + \mu_i - C)\mathbf{z}_i^r ,$$

$$\mathbf{w}_r^* = \frac{1}{\gamma}\sum_{i \in T_r}(\alpha_i^* + \mu_i^* - C)\mathbf{z}_i^r ,$$

where the coefficients α_i, α_i^*, μ_i and μ_i^* are solution of the following dual optimization problem:

$$\begin{aligned}
\min_{\alpha,\alpha^*,\beta,\beta^*} \quad & \varepsilon\sum_{i=1}^{n}(\alpha_i^*+\alpha_i)-\sum_{i=1}^{n}(\alpha_i^*-\alpha_i)y_i+\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)(\mathbf{z}_i\mathbf{z}_j) \\
& +\frac{1}{2\gamma}\sum_{r=1}^{t}\sum_{i,j\in T_r}(\alpha_i+\beta_i-C)(\alpha_j+\beta_j-C)(\mathbf{z}_i^r\mathbf{z}_j^r) \\
& +\frac{1}{2\gamma}\sum_{r=1}^{t}\sum_{i,j\in T_r}(\alpha_i^*+\beta_i^*-C)(\alpha_j^*+\beta_j^*-C)(\mathbf{z}_i^r\mathbf{z}_j^r) \\
\text{subject to:} \quad & \sum_{i=1}^{n}(\alpha_i^*-\alpha_i)=0 \qquad (5.3) \\
& \sum_{i\in T_r}(\alpha_i+\beta_i)=C\,|T_r|,\ r=1,\mathrm{K}\ ,t \\
& \sum_{i\in T_r}(\alpha_i^*+\beta_i^*)=C\,|T_r|,\ r=1,\mathrm{K}\ ,t \\
& \alpha_i\ge 0,\alpha_i^*\ge 0,\beta_i\ge 0,\beta_i^*\ge 0,\ i\in T_r, r=1,\mathrm{K}\ ,t
\end{aligned}$$

5.3 SVM+MTL Regression

Similar to SVM+, SVM+MTL also makes good use of group information. However, the goal of this approach is to estimate *t* regression models (one model per group). So instead of incorporating group information into slack variables, SVM+MTL incorporates group information into estimated regression functions.

We specify the following parameterization for t regression models:

$$f_r(\mathbf{x})=(\phi_z(\mathbf{x}),\mathbf{w})+b+(\phi_{z_r}(\mathbf{x}),\mathbf{w}_r)+d_r, r=1,...,t \qquad (5.4)$$

Here $(\phi_z(\mathbf{x}),\mathbf{w})+b$ is the common estimation function while $(\phi_{z_r}(\mathbf{x}),\mathbf{w}_r)+d_r$ is the unique correction function for each group. The proposed method SVM+MTL formulation solves the following optimization problem:

$$\min_{\substack{\mathbf{w},\mathbf{w}_r,b,\mathbf{d}_r \\ r=1,\mathrm{K},t}} \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + \frac{\gamma}{2}\sum_{r=1}^{t}(\mathbf{w}_r,\mathbf{w}_r) + C\sum_{r=1}^{t}\sum_{i\in T_r}(\xi_i^r + \xi_i^{r*})$$

$$\text{subject to: } y_i - ((\mathbf{w}\cdot\mathbf{z}_i) + b + (\mathbf{w}_r\cdot\mathbf{z}_i^r) + d_r) \le \varepsilon + \xi_i^{r*} \qquad (5.5)$$

$$((\mathbf{w}\cdot\mathbf{z}_i) + b + (\mathbf{w}_r\cdot\mathbf{z}_i^r) + d_r) - y_i \le \varepsilon + \xi_i^r$$

$$\xi_i^r \ge 0, \xi_i^{r*} \ge 0,\ i \in T_r, r = 1,\mathrm{K},t$$

The dual form of the above optimization problem is as follows:

$$\min_{\alpha,\alpha^*} \varepsilon\sum_{i=1}^{n}(\alpha_i^* + \alpha_i) - \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)y_i + \frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)(\mathbf{z}_i\cdot\mathbf{z}_j)$$

$$+ \frac{1}{2\gamma}\sum_{r=1}^{t}\sum_{i,j\in T_r}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)(\mathbf{z}_i^r\cdot\mathbf{z}_j^r) \qquad (5.6)$$

$$\text{subject to: } \sum_{i\in T_r}(\alpha_i^* - \alpha_i) = 0,\ r = 1,\mathrm{K},t$$

$$0 \le \alpha_i \le C,\ 0 \le \alpha_i^* \le C,\ i = 1,\mathrm{K},n$$

Based on KKT condition, we can express $\mathbf{w}$, $\mathbf{w}_r$ in terms of training samples:

$$\mathbf{w} = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)\mathbf{z}_i,$$

$$\mathbf{w}_r = \frac{1}{\gamma}\sum_{i\in T_r}(\alpha_i^* - \alpha_i)\mathbf{z}_i^r.$$

Thus, $f_r(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i)(\mathbf{z}_i,\phi_z(\mathbf{x})) + b + \frac{1}{\gamma}\sum_{i\in T_r}(\alpha_i^* - \alpha_i)(\mathbf{z}_i^r,\phi_{z^r}(\mathbf{x})) + d_r, r = 1,\mathrm{K},t$.

5.4 Empirical Comparisons

We assume that training data for our regression problems can be partitioned (in a meaningful way) into several group. Similar to the classification setting in Chapter 3, we can identify several learning (regression) approaches for utilizing the group information, as described next:

- *Single SVM* inductive model which estimates standard SVM regression by pooling together training samples from different groups (i.e. group information is ignored);
- *multiple SVM* approach where a separate SVM regression model is estimated for each group (independently);
- *SVM+* approach where a single regression model, utilizing available group information, is estimated from all data;
- *SVM+MTL* implementing multi-task learning, which estimates several related regression models.

Various approaches for incorporating group data into learning process are shown in Figure 5.2. Theoretically, one can expect more sophisticated modeling approaches (utilizing the group information), i.e., SVM+ and SVM+MTL, to yield better generalization than single inductive SVM and multiple (independent) SVM's, respectively. In practice, the trade-off is not so clear, because more advanced approaches (SVM+ and SVM+MTL) have more tunable parameters (than standard SVM), and their potential advantages can be easily offset by more complex model selection.

Next we consider tunable parameters for various learning approaches:

- Single SVM regression: parameters C, ε (width of insensitive zone) and σ (if RBF kernel is used);
- Multiple SVMs: parameters C, ε and σ (if RBF kernel is used) for each task;

- SVM+ regression, where same kernel as in standard SVM is used for the common space, and RBF kernel is used for correcting space, requires following parameters C, ε and σ_{common} (as in standard SVM), γ and $\sigma_{correction}$ (RBF width);

- SVM+MTL regression: requires following parameters C, ε and σ_{common} (as in standard SVM), γ and $\sigma_{correction}$ (RBF width parameter).

Clearly, application of SVM+ and SVM+MTL regression requires practical strategies for tuning parameters of these methods for a given data set. Next we discuss such a strategy that will be used in empirical comparisons presented later in Section 4. For standard SVM regression, we use analytic prescription for parameters C following (Cherkassky and Ma, 2004) and resampling for ε and σ. The same approach is used for multiple SVMs, where parameters C, ε and σ are selected for each group (task). For SVM+ and SVM+MTL, parameters are tuned in a two-stage manner: first set parameters parameters C, ε and σ_{common} same as for standard SVM regression, and second, select parameters γ and $\sigma_{correction}$ via resampling. Empirical comparisons presented next use 5-fold cross-validation for estimating γ and σ.

Next we describe empirical comparisons of various modeling approaches such as single SVM (sSVM), multiple SVM (mSVM), SVM+ and SVM+MTL. All comparisons for synthetic data use linear parameterization for sSVM and mSVM, and RBF(Gaussian) kernel is used for real data. The common estimation space for SVM+ and SVM+MTL use linear kernel for synthetic data and RBF kernel for real data while the unique correction space use RBF kernel. For each modeling method, we present the predicted

mean squared error (MSE) for each of the five folds, as well as the mean and standard deviation of the MSEs. Both real data and synthetic data are used.

- **Real data**

All comparisons of real data follow the following experimental procedure:

(a) Select a group variable (from a list of input variables).

(b) Partition available data into several groups (tasks) corresponding to different values (or range of values) of group variable. Each group should be roughly of similar size.

(c) Within each group, order data samples by increasing value of the group variable.

(d) For estimating prediction error of a particular method, use 5-fold cross-validation, so that 80% of data samples are used for training and 20% of the data are used as test data. Note that conditions (b) and (c) ensure that each fold has approximately equal number of samples from all groups (tasks).

(e) For each training fold, parameter tuning (model selection) for different methods is performed as specified in Section 3. That is, parameter C is estimated first following [3], and then parameters ε, γ and σ are tuned via resampling within the training fold.

A. Boston Housing Dataset

This dataset is available at UCI machine learning repository. It includes 14 variables (13 continuous and 1 Boolean) and 506 instances. The goal is to estimate the median value of owner-occupied homes in $1000's from 13 attributes. We present two different

comparisons for this dataset. *First,* variable 'RAD' is selected to separate data into 3 groups: group 1($RAD < 5$, 192 instances), group 2($5 \le RAD < 7.5$, 158 instances) and group 3($RAD \ge 7.5$, 156 instances). *Second,* we separate data into 3 groups by another variable 'DIS': group 1($DIS < 2.5$, 188 instances), group 2($2.5 \le DIS < 4.5$, 163 instances) and group 3($DIS \ge 4.5$, 155 instances). Therefore, sSVM, mSVM, SVM+ and SVM+MTL all use 13 attributes for prediction. Possible choices of parameters for SVM+ and SVM+MTL are: $\gamma = [10\ 1\ 0.1\ 0.01\ 0.001]$, $\sigma = [0.25\ 0.5\ 1\ 3\ 4]$. Results are shown in Table 5.1 and Table 5.2.

B. Auto MPG Dataset

This is another dataset from UCI machine learning repository. There are 398 instances, each of which has 8 input variables and 1 output variable (mpg). The input variable 'horsepower' has 6 missing values. After removing 6 samples with missing values, we are left with 392 samples used for modeling. The goal is to estimate the real-valued output 'mpg' for each car using 7 input variables (the input variable 'car name' is not used for modeling). We choose variable 'cylinders' to separate the data into 3 groups: group1($cylinders < 6$, with 206 instances), group2 ($cylinders = 6$, with 83 instances), group3($cylinders > 6$, with 103 instances). Possible choices of parameters for SVM+ and SVM+MTL are: $\gamma = [10\ 1\ 0.1\ 0.01\ 0.001]$, $\sigma = [0.25\ 0.5\ 1\ 3\ 4]$. Modeling results are shown in Table 5.3. In both cases, the SVM+MTL method shows some improvement in MSE prediction error, over all other methods.

- **Synthetic data**

This data set was artificially generated, in order to illustrate the effect of training set size and the noise level, on relative performance of different learning methods.

Synthetic data set was generated as follows:

(1) Let number of input features be $d = 30$, and number of tasks(groups) be $t = 3$.

(2) Generate $\mathbf{x} \in R^{30}$ with each component $x_i \sim uniform(0,1), i = 1,...,30$.

(3) The coefficient vectors of three (linear) regression tasks are specified as:

$\beta_1 = [1,1,2,3,3,1,1,1,1,0,2,0,2,2,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0]$,
$\beta_2 = [1,1,2,3,3,1,1,1,1,0,2,0,2,2,0,0,0,0,2,0,2,0,0,0,0,0,0,0,0,0]$,
$\beta_3 = [1,1,2,3,3,1,1,1,0,1,0,0,3,0,0,2,0,2,0,2,0,0,0,0,0,0,0,0,0,0]$.

(4) For each task and each data vector, $y = \beta_i \mathbf{x} + n$, where n is Gaussian noise with

$\sigma_{noise} = 1$.

For each task, we generate N samples for training, N samples for validation, and 1000 samples for test, and repeat the experiment 5 times. We follow the ways of tuning parameters earlier for real data except that γ and σ for SVM+ and SVM+MTL are now tuned using validation data. Possible choices of parameters for SVM+ and SVM+MTL are $\gamma = [100\ 10\ 1\ 0.1\ 0.01]$, $\sigma = [0.25\ 0.5\ 1\ 4\ 6\ 8]$. Comparison results, for different values of N (number of training samples per task), are shown in Table 5.4 and Table 5.5. Note that SVM+MTL shows superior prediction accuracy vs other learning approaches.

In the next experiment, we reduced the input dimensionality ($d = 20$) and noise level($\sigma_{noise} = 0.5$), and increased the sample size used for training and validation ($N = 100$).

Also, new coefficient vectors used to generate target functions for three tasks are specified as follows:

$$\beta_1 = [1,1,2,3,3,1,1,1,1,0,4,0,4,4,0,0,0,0,0,0],$$

$$\beta_2 = [1,1,2,3,3,1,1,1,0,4,0,2,4,0,0,0,0,0,0,0],$$

$$\beta_3 = [1,1,2,3,3,1,1,0,1,0,0,4,0,0,4,0,0,0,0,0].$$

Comparison results shown in Table 6 indicate that SVM+MTL underperformed mSVM. Careful examination of results in Tables 4.4-4.6 makes it possible to relate performance of different learning approaches to statistical characteristics of synthetic data sets, as discussed below:

- Synthetic data set 1 (see Table 5.4) is small and very noisy, so one can expect that methods emphasizing utilization of group information (such as SVM+ and SVM+MTL) yield better prediction performance;
- Synthetic data set 2 (see Table 5.5) has more training samples (N=50 vs 20 for set 1). Therefore, separating training data into several independent groups as in mSVM approach can now be beneficial, and this accounts for relative improvement in the prediction accuracy of mSVM;

Synthetic data set 3 (see Table 5.6) has largest number of training samples, and very low noise level. Hence, we can expect that independent estimation of individual regression models, as in mSVM, would yield the best prediction accuracy.

5.5 Conclusion

This chapter presents new methodology called SVM+MTL regression, for utilizing

group information in regression problems. This approach extends original Vapnik's SVM+ regression technology to multi-task learning. Empirical comparisons using several data sets show that the proposed SVM+MTL regression can provide significant improvement in prediction accuracy vs standard inductive SVM regression. Further, we discussed several approaches for utilizing the group information available in real-life data sets, including standard inductive SVM, multiple SVMs, SVM+ and SVM+MTL. Whereas presented empirical comparisons may suggest the advantages of SVM+MTL, we strongly warn against making such over-reaching conclusions. Relative performance of learning methods is always strongly affected by the properties of application data at hand. To this end, we also presented a synthetic data set where the proposed method shows inferior generalization performance. New learning settings, such as SVM+ regression and SVM+MTL regression, are more complex than standard SVM, and have more tuning parameters. Therefore, practical application of these new learning methodologies requires robust strategies for model selection. This paper describes such a practical strategy that combines analytic tuning of two parameters for standard SVM regression, followed by resampling approach for tuning parameters specific to SVM+ and SVM+MTL regression formulations.

Table 5.1 Test MSE for Boston housing dataset (group variable: RAD)

Folds	1	2	3	4	5	Mean(st.dev)
sSVM	8.9	26.1	8.5	5.9	10.9	12.1(8.0)
mSVM	12.1	27.2	10.4	6.2	15.1	14.2(7.9)
SVM+	8.9	23.5	9.5	6.1	8.8	11.4(6.9)
SVM+MTL	**7.6**	**15.6**	**8.0**	**4.9**	**8.7**	**9.0(4.0)**

Table 5.2 Test MSE for Boston housing dataset (group variable: DIS)

Folds	1	2	3	4	5	Mean(st.dev)
sSVM	8.9	8.3	11.1	9.0	18.4	11.1(4.2)
mSVM	10.2	8.9	10.3	11.1	20.1	12.1(4.5)
SVM+	8.1	8.7	10.7	**7.9**	**16.5**	10.4**(3.6)**
SVM+MTL	**7.1**	**8.2**	**8.6**	8.4	17.0	**9.9**(4.0)

Table 5.3 Test MSE for auto mpg dataset (group variable: cylinders)

Folds	1	2	3	4	5	Mean(st.dev)
sSVM	6.4	**6.8**	5.9	10.6	6.6	7.3(1.9)
mSVM	5.9	7.4	6.5	10.8	8.2	7.8(1.9)
SVM+	6.7	6.9	5.8	10.3	6.6	7.3**(1.8)**
SVM+MTL	**5.5**	6.9	**5.7**	**10.3**	**4.5**	**6.6**(2.2)

Table 5.4 Test MSE for synthetic data set 1
($d = 30$ **,** $N = 20$ **,** $\sigma_{noise} = 1$ **)**

Trials	1	2	3	4	5	Mean(st.dev)
sSVM	4.2	3.7	4.2	4.0	3.8	4.0**(0.21)**
mSVM	5.1	4.9	4.5	5.1	4.9	4.9(0.27)
SVM+	4.3	3.3	3.5	3.9	4.0	3.8(0.37)
SVM+MTL	**2.9**	**2.7**	**2.9**	**3.0**	**3.4**	**3.0**(0.25)

Table 5.5 Test MSE for synthetic data set 2
($d = 30$ **,** $N = 50$ **,** $\sigma_{noise} = 1$ **)**

Trials	1	2	3	4	5	Mean(st.dev)
sSVM	2.5	2.4	2.6	2.5	2.5	2.5**(0.08)**
mSVM	2.7	1.8	2.8	2.1	2.2	2.3(0.44)
SVM+	2.5	2.4	2.4	3.4	3.0	2.7(0.44)
SVM+MTL	**1.9**	**1.7**	**1.8**	**1.7**	**1.9**	**1.8**(0.11)

Table 5.6 Test MSE for synthetic data set 3
($d = 20$ **,** $N = 100$ **,** $\sigma_{noise} = 0.5$ **)**

Trials	1	2	3	4	5	Mean(st.dev)
sSVM	7.8	7.8	8.1	7.8	7.6	7.8(0.15)
mSVM	**0.3**	**0.4**	**0.3**	**0.3**	**0.3**	**0.3(0.01)**
SVM+	7.8	7.9	7.5	7.6	7.7	7.7(0.13)
SVM+MTL	1.0	1.0	1.1	1.1	1.1	1.1(0.06)

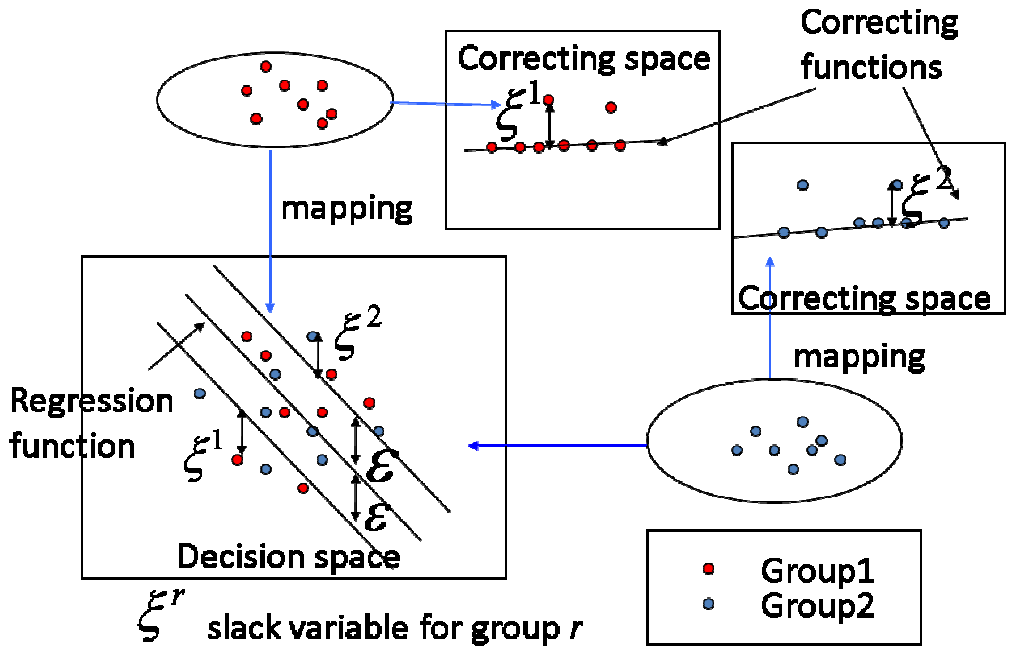

Figure 5.1 SVM+ maps data simultaneously into decision space and correcting spaces. Regression function is found in decision space. Slack variables are represented by correcting functions which are defined in correcting space.

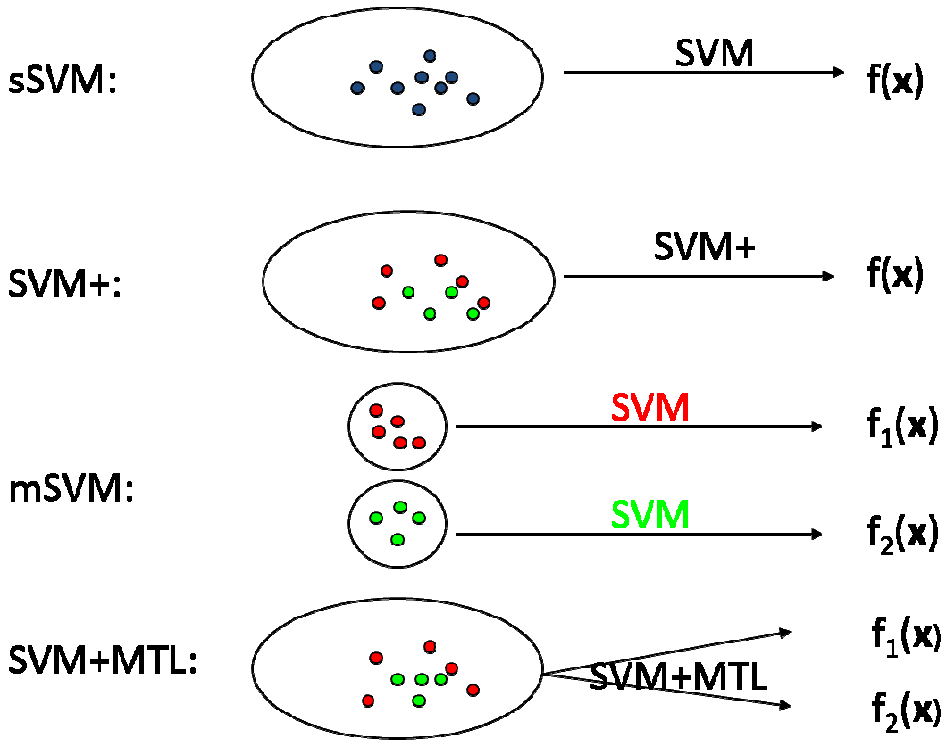

Figure 5.2 Different ways of using group information in learning: (a) sSVM ~ Single SVM regression, (b) SVM+ regression, (c) mSVM ~ multiple (independent) SVMs, and (d) SVM+MTL ~ proposed SVM+ Multi-Task Learning.

Chapter 6 Generalized SMO Training for SVM+MTL

This chapter describes our proposed Generalized Sequential Minimal Optimization (GSMO) for SVM+MTL. Empirical results show that our GSMO for SVM+MTL is much faster than existing implementations based on CVX optimization package (Liang and Cherkassky, 2008; Liang, Cai and Cherkassky, 2009). The complexity of GSMO for SVM+MTL is comparable to SMO for standard SVM.

6.1 Introduction

In previous chapters, we discussed several SVM based non-inductive methodologies such as SVM+ and SVM+MTL. These new methods effectively utilize extra structure information (i.e. group information) during training, which result in improved generalization. Large quadratic programming (QP) problems need to be solved to train SVM+ and SVM+MTL. Therefore, we need to find efficient training algorithms for SVM+ and SVM+MTL. Since SVM+ and SVM+MTL are generalized from SVM, their formulations are quite similar. Therefore, one way of training SVM+ and SVM+MTL is to generalize training algorithm of SVM for SVM+ and SVM+MTL.

The computational complexity of solving the QP problem in SVM training grows as $O(n^3)$, where n is sample size (Tsang, Kwok and Cheung, 2005). This is slow when sample size n is large. One of the widely used fast algorithms to train SVM is the SMO algorithm of John Platt (1998). This algorithm solves the dual problem of SVM in the iterative way. It breaks the large QP problem into a series of sub-QP problems of the

smallest possible size. Each sub-QP problem has only two variables and analytical solution can be found for this small QP problem, which makes the training much faster. The decision which pair of variables should be optimized at the current iteration is done by some working set selection rule (Platt, 1998 and Fan, Chen and Lin, 2005). Smola et al. proposed SMO for SVM regression (1998). Shevade et al. made some improvements to SMO by employing two threshold parameters (2000). Recently, a Generalized SMO (GSMO) was developed for SVM+ under LUPI setting (Pechyony et al. 2010). Motivated by standard SMO and GSMO for SVM+, we propose the Generalized SMO (GSMO) for SVM+MTL.

6.2 SMO for SVM

Given finite training data $(\mathbf{x}_i, y_i), i = 1,\text{K}\ ,n,\ \ \mathbf{x} \in R^d$ and $y \in \{+1,-1\}$, SVM estimates decision function $f(\mathbf{x}) = \mathbf{w}\mathbf{g}\mathbf{z} + b$ by solving the following problem:

$$\begin{aligned} &\min_{\mathbf{w},b} \ \ \frac{1}{2}\| \mathbf{w} \|^2 + C\sum_{i=1}^{n} \xi_i \\ &\text{subject to:}\ \ y_i(\mathbf{w}\mathbf{g}\mathbf{z}_i + b) \geq 1 - \xi_i,\ i = 1,\text{K}\ ,n. \end{aligned} \tag{6.1}$$

where $\mathbf{z}_i, i = 1,\text{K}\ n$ is feature map of $\mathbf{x}_i, i = 1,\text{K}\ n$.

A common way of solving the primal problem (6.1) is to its dual problem:

$$\begin{aligned} &\min_{\alpha} \ \ l(\alpha) = -\sum_{i=1}^{n} \alpha_i + \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i, \mathbf{x}_j) \\ &\text{subject to:}\ \ \sum_{i=1}^{n} \alpha_i y_i = 0,\ \ 0 \leq \alpha_i \leq C,\ i = 1,\text{K}\ ,n. \end{aligned} \tag{6.2}$$

The complexity of solving this large QP problem is $O(n^3)$. There are several successful

decomposition algorithms which try to solve the dual problem by breaking the large QP problem into a series of small QP problems. An extreme case is the SMO algorithm (Platt, 1998) , which restricts the small QP problems to have only two variables. Then in each iteration a simple two-variable optimization problem can be solved analytically. This method is sketched in algorithm 6.1.

Algorithm 6.1 SMO decomposition method (Platt, 1998)

---

1. Set α as the initial feasible solution.
2. If α is an optimal solution of (6.2), stop. Otherwise, find a working set with only two variables α_p and α_q.
3. Solve the following sub-problem with two variables α_p and α_q:

$$\min_{\alpha_p,\alpha_q} \frac{1}{2}\alpha_p^2 y_p^2(\mathbf{z}_p \cdot \mathbf{z}_p) + \frac{1}{2}\alpha_q^2 y_q^2(\mathbf{z}_q \cdot \mathbf{z}_q) + \alpha_p \alpha_q y_p y_q(\mathbf{z}_p \cdot \mathbf{z}_q)$$
$$+\alpha_p(\frac{1}{2}\sum_{i\neq p,q}\alpha_i y_p y_i(\mathbf{z}_i \cdot \mathbf{z}_p) - 1) + \alpha_q(\frac{1}{2}\sum_{i\neq p,q}\alpha_i y_q y_i(\mathbf{z}_i \cdot \mathbf{z}_q) - 1) + C_1 \quad (6.3)$$

subject to: $\alpha_p y_p + \alpha_q y_q = C_2$

$$0 \le \alpha_p \le C,\ 0 \le \alpha_p \le C,$$

where C_1 and C_2 are constants.

4. Update α with new α_p and α_q from step 3, go to step 2.

The decomposition method suffers from slow convergence for difficult problems since only few components are updated per iteration. A good working set selection method in step 2 of algorithm 6.1 is very critical for the speed of convergence. There

exist several good working set selection methods (Keerthi et al. 1999; Fan, Chen and Lin, 2005), which can reduce the number of iterations. Some of the methods use the first order (i.e. gradient) information of the objective function. One successful method is the working set selection via the maximal violating pair as follows (Keerthi et al. 1999):

$$\begin{aligned} &\text{select } p \in \arg\max_i \{-y_i \nabla l(\alpha)_i \mid i \in I_{\text{up}}(\alpha)\}, \\ &\quad q \in \arg\min_i \{-y_i \nabla l(\alpha)_i \mid i \in I_{\text{low}}(\alpha)\}, \\ &\text{where } I_{\text{up}}(\alpha) \equiv \{i \mid \alpha_i < C, y_i = 1 \text{ or } \alpha_i > 0, y_i = -1\}, \\ &\quad I_{\text{low}}(\alpha) \equiv \{i \mid \alpha_i < C, y_i = -1 \text{ or } \alpha_i > 0, y_i = 1\}. \end{aligned} \tag{6.4}$$

Here $\nabla l(\alpha)$ is the gradient of objective function $l(\alpha)$ in dual problem (6.2). Suppose we have a pair p, q and $p \in I_{\text{up}}(\alpha)$, $q \in I_{\text{low}}(\alpha)$, it can be derived through KKT condition that

$$-y_p \nabla l(\alpha)_p \le -y_q \nabla l(\alpha)_q \tag{6.5}$$

If equation (6.5) is violated, then the pair p and q is called a violating pair. Moreover, if p and q are selected by (6.4), then this violating pair is a maximal violating pair. It is shown (Keerthi et al., 1999) that this working set selection leads to a much faster SMO than Platt's original SMO.

Another successful method for working set selection uses the second order information of the objective function. Empirical results showed that this method is much better than the maximal violating pair selection. Given a pair of variables α_p and α_q, the second order information yields more accurate representation for the change of the quadratic function $l(\alpha)$:

$$\Delta l(\alpha)_{pq} = l(\alpha + \Delta\alpha) - l(\alpha) = \nabla l(\alpha)^T \Delta\alpha + \frac{1}{2}\Delta\alpha^T \nabla^2 l(\alpha)\Delta\alpha \qquad (6.6)$$

where $\Delta\alpha_i = 0$ for all $i \neq p, q$. The ideal working set selection using second order information should be the violating pair that minimizes the change of objective function as in equation (6.6). However, it is impractical to check all $\binom{n}{2}$ pairs in order to find an optimal pair. A viable implementation of using second order information involves checking just a few pairs selected using heuristic rules proposed in (Fan, Chen and Lin, 2005). The pair of p and q is selected in two steps, p is selected first, then q is selected:

$$\begin{aligned} &p \in \arg\max_i \{-y_i \nabla l(\alpha)_i \mid i \in I_{\text{up}}(\alpha)\}, \\ &q \in \arg\min_i \{\min_{\Delta\alpha_i, \Delta\alpha_p} \Delta l(\alpha)_{pi} \mid i \in I_{\text{low}}(\alpha), -y_i \nabla l(\alpha)_i < -y_p \nabla l(\alpha)_p\}. \end{aligned} \qquad (6.7)$$

This working set selection checks only $O(n)$ possible pairs for selecting q . Let $a_{ij} = (\mathbf{z}_i \mathbf{z}_i) + (\mathbf{z}_j \mathbf{z}_j) - 2(\mathbf{z}_i \mathbf{z}_j)$, $v_{ij} = -y_i \nabla l(\alpha)_i + -y_j \nabla l(\alpha)_j$ and τ be a predefined constant. Then the simplified procedure has the following form (Fan, Chen and Lin, 2005):

$$\begin{aligned} &p \in \arg\max_i \{-y_i \nabla l(\alpha)_i \mid i \in I_{\text{up}}(\alpha)\}, \\ &q \in \arg\min_i \{-\frac{v_{pi}^2}{\bar{a}_{pi}} \mid i \in I_{\text{low}}(\alpha), -y_i \nabla l(\alpha)_i < -y_p \nabla l(\alpha)_p\}, \end{aligned} \qquad (6.8)$$

Where

$$\bar{a}_{pi} \equiv \begin{cases} a_{pi} & \text{if } a_{pi} > 0, \\ \tau & \text{otherwise.} \end{cases}$$

It can be shown that working set selection (6.8) leads to the convergence after a finite number of iterations (Fan, Chen and Lin, 2005). Therefore, our generalized SMO will use

a similar working set selection strategy.

6.3 GSMO for SVM+MTL Classification

Motivated by the success of SMO, we design the GSMO for SVM+MTL. Formally, suppose the training data can be naturally partitioned into t related groups. Let us denote the indices from group r by $T_r = \{i_{r_{n_1}}, \mathrm{K}, i_{r_{n_r}}\}, r = 1, \mathrm{K}, t$. Then all training samples can be represented as: $\{\{X_r, Y_r\}, r = 1, \mathrm{K}, t\}, \{X_r, Y_r\} = \{\{\mathbf{x}_{r_{n_1}}, y_{r_{n_1}}\}, \mathrm{K}, \{\mathbf{x}_{r_{n_r}}, y_{r_{n_r}}\}\}, \mathbf{x} \in R^d, y \in \{-1, +1\}$. SVM+MTL estimates t related classifiers: $f_r(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{z}) + b + (\mathbf{w}_r \cdot \mathbf{z}^r) + d_r,\ r = \{1, ..., t\}$ by solving the following optimization problem:

$$\begin{aligned} \min_{\substack{\mathbf{w}, b, \mathbf{w}_r, d_r \\ r=1,\mathrm{K},t}} \quad & \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + \frac{\gamma}{2}\sum_{r=1}^{t}(\mathbf{w}_r \cdot \mathbf{w}_r) + C\sum_{r=1}^{t}\sum_{i \in T_r}\xi_i^r \\ \text{subject to:} \quad & y_i((\mathbf{w} \cdot \mathbf{z}_i) + b + (\mathbf{w}_r \cdot \mathbf{z}_i^r) + d_r) \geq 1 - \xi_i^r, \qquad (6.9) \\ & \xi_i^r \geq 0,\ i \in T_r,\ r = 1, \mathrm{K}, t. \end{aligned}$$

Similar to SMO for SVM, our GSMO for SVM+MTL solves the dual problem of (6.9):

$$\begin{aligned} \min_{\alpha} \quad & l_{\mathrm{mtl}}(\alpha) = -\sum_{i=1}^{n}\alpha_i + \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j (\mathbf{z}_i \cdot \mathbf{z}_j) \\ & + \frac{1}{2\gamma}\sum_{r=1}^{t}\sum_{i,j \in T_r}\alpha_i\alpha_j y_i y_j (\mathbf{z}_i^r \cdot \mathbf{z}_j^r) \qquad (6.10) \\ \text{subject to:} \quad & \sum_{i \in T_r}\alpha_i y_i = 0,\ 0 \leq \alpha_i \leq C,\ i \in T_r, r = 1, \mathrm{K}, t. \end{aligned}$$

Compared with dual problem of SVM (6.2), the dual problem of SVM+MTL (6.10) has more equality constraints, one equality constraint for each group. Therefore, in SVM+MTL, the choice of two variable working set $\{\alpha_p, \alpha_q\}$ must satisfy additional constraints. Since there is one equality constraint for each group, the two variables α_p

and α_q should come from same group. The working set selection (6.7) for SVM uses a two step heuristic. We design the following working set selection strategy for SVM+MTL. For each group r, two variables α_{p_r} and α_{q_r} are selected by method similar to (6.7), which results in *t* working pairs for *t* groups. Finally, the pair which gives the maximum decrease of objective function in (6.10) is selected.

Suppose α_p and α_q are a pair of variables selected from a group r, the two variable sub-problem for SVM+MTL is:

$$\begin{aligned}
\min_{\alpha_p,\alpha_q}\ & \frac{1}{2}\alpha_p^2 y_p^2((\mathbf{z}_p \bullet \mathbf{z}_p)+\frac{1}{\gamma}(\mathbf{z}_p^r \bullet \mathbf{z}_p^r))+\frac{1}{2}\alpha_q^2 y_q^2((\mathbf{z}_q \bullet \mathbf{z}_q)+\frac{1}{\gamma}(\mathbf{z}_q^r \bullet \mathbf{z}_q^r)) \\
& +\alpha_p\alpha_q y_p y_q((\mathbf{z}_p \bullet \mathbf{z}_q)+\frac{1}{\gamma}(\mathbf{z}_p^r \bullet \mathbf{z}_q^r)) \\
& +\alpha_p(\frac{1}{2}\sum_{i\neq p,q}\alpha_i y_p y_i(\mathbf{z}_i \bullet \mathbf{z}_p)+\frac{1}{2\gamma}\sum_{i\neq p,q,\ i\in T_r}\alpha_i y_p y_i(\mathbf{z}_i^r \bullet \mathbf{z}_p^r)-1) \\
& +\alpha_q(\frac{1}{2}\sum_{i\neq p,q}\alpha_i y_q y_i(\mathbf{z}_i \bullet \mathbf{z}_q)+\frac{1}{2\gamma}\sum_{i\neq p,q,\ i\in T_r}\alpha_i y_q y_i(\mathbf{z}_i^r \bullet \mathbf{z}_q^r)-1)+C_1 \\
\text{subject to: }\ & \alpha_p y_p+\alpha_q y_q = C_2 \\
& 0\le\alpha_p\le C,\ 0\le\alpha_q\le C.
\end{aligned} \tag{6.11}$$

Similar to the sets $I_{\text{up}}(\alpha)$ and $I_{\text{low}}(\alpha)$ for SVM, for each group r, SVM+MTL has the following sets:

$$\begin{aligned}
I_{\text{up}}^r(\alpha) &\equiv \{i \mid \alpha_i < C, y_i = 1 \text{ or } \alpha_i > 0, y_i = -1,\ \ i \in T_r\}, \\
I_{\text{low}}^r(\alpha) &\equiv \{i \mid \alpha_i < C, y_i = -1 \text{ or } \alpha_i > 0, y_i = 1\ \ i \in T_r\}.
\end{aligned}$$

If $p \in I_{\text{up}}^r(\alpha)$ and $q \in I_{\text{low}}^r(\alpha)$, we have the following equation:

$$-y_p \nabla l_{\text{mtl}}(\alpha)_p \le -y_q \nabla l_{\text{mtl}}(\alpha)_q . \tag{6.12}$$

If (6.12) is violated, p and q form a violating pair. Equation (6.12) can be derived

through KKT condition of (6.10). Suppose $i \in T_r$, a vector α is a stationary point of (6.10) if and only if there is a number u and two nonnegative vectors λ and μ such that

$$\begin{aligned} &\nabla l_{\text{mtl}}(\alpha)_i + uy_i = \lambda_i - \mu_i, \\ &\lambda_i \alpha_i = 0, \mu_i (C - \alpha_i) = 0, \lambda_i \geq 0, \mu_i \geq 0. \end{aligned} \tag{6.13}$$

Equation (6.13) can be rewritten as

$$\begin{aligned} &\nabla l_{\text{mtl}}(\alpha)_i + uy_i \geq 0 \ \text{ if } \ \alpha_i < C, \\ &\nabla l_{\text{mtl}}(\alpha)_i + uy_i \leq 0 \ \text{ if } \ \alpha_i > 0. \end{aligned} \tag{6.14}$$

Rewriting (6.14) as the following form:

$$\begin{aligned} &-y_i \nabla l_{\text{mtl}}(\alpha)_i \leq u, \ \forall i \in I_{\text{up}}^r(\alpha), \\ &-y_i \nabla l_{\text{mtl}}(\alpha)_i \geq u, \ \forall i \in I_{\text{low}}^r(\alpha). \end{aligned} \tag{6.15}$$

We can see that condition (6.12) is satisfied.

A. Working set selection using second order information

Next we discuss the working set selection using the second order information for SVM+MTL. Since $l_{\text{mtl}}(\alpha)$ is quadratic, when the variable α changes to $\alpha + \Delta\alpha$, the change of $l_{\text{mtl}}(\alpha)$ will be as follows:

$$\begin{aligned} \Delta l_{\text{mtl}}(\alpha) &= l_{\text{mtl}}(\alpha + \Delta\alpha) - l_{\text{mtl}}(\alpha) \\ &= \nabla_{\text{mtl}}(\alpha)^T \Delta\alpha + \frac{1}{2} \Delta\alpha^T \nabla^2 l_{\text{mtl}}(\alpha) \Delta\alpha. \end{aligned} \tag{6.16}$$

Suppose two variables α_p and α_q from same group r are selected to be optimized, then $\Delta\alpha_i = 0 \forall i \neq p, q.$ The change $\Delta l_{\text{mtl}}(\alpha)$ in (6.16) can be written as:

$$\begin{aligned}\Delta l_{\text{mtl}}(\alpha)_{pq} &= \nabla l_{mtl}(\alpha)_p \Delta\alpha_p + \nabla l_{mtl}(\alpha)_q \Delta\alpha_q \\ &+ \frac{1}{2}\nabla^2 l_{\text{mtl}}(\alpha)_{pp}\Delta\alpha_p^2 + \frac{1}{2}\nabla^2 l_{\text{mtl}}(\alpha)_{qq}\Delta\alpha_q^2 \\ &+ \nabla^2 l_{\text{mtl}}(\alpha)_{pq}\Delta\alpha_p\Delta\alpha_q\end{aligned} \quad (6.17)$$

With respect to variables $\Delta\alpha_p$ and $\Delta\alpha_q$, the minimum of (6.17) can be found. Denote this minimum as $\Delta l_{\text{mtl}}(\alpha)_{pq}^{\min}$, the working set selection using second order information tries to find working set α_p and α_q from group r such that $\Delta l_{\text{mtl}}(\alpha)_{pq}^{\min}$ is minimum. Therefore, the working set $\{p,q\}$ from same group r can be found as follows:

$$\{p,q\} \in \arg\min_{p,q\in T_r, 0\le r\le t} \Delta l_{\text{mtl}}(\alpha)_{pq}^{\min} \quad (6.18)$$

Where

$$\begin{aligned}\Delta l_{\text{mtl}}(\alpha)_{pq}^{\min} = &\min_{\Delta\alpha_p,\Delta\alpha_q} \Delta l_{\text{mtl}}(\alpha)_{pq} \\ &\text{subject to: } y_p\Delta\alpha_p + y_q\Delta\alpha_q = 0, \\ &\Delta\alpha_i \ge 0, \text{ if } \alpha_i = 0, i = p,q, \\ &\Delta\alpha_i \le 0, \text{ if } \alpha_i = C, i = p,q.\end{aligned} \quad (6.19)$$

To solve the set in (6.18) by exhaustively check $\binom{n_r}{2}$ pairs of each group r is not pratical. Therefore, a working set selection method similar to (6.7) should be used in group r for SVM+MTL as well:

$$\begin{aligned}&p \in \arg\max_i \{-y_i \nabla l_{\text{mtl}}(\alpha)_i \mid i \in I_{\text{up}}^r(\alpha)\}, \\ &q \in \arg\min_i \{\Delta l_{\text{mtl}}(\alpha)_{pi}^{\min} \mid i \in I_{\text{low}}^r(\alpha), -y_i \nabla l_{\text{mtl}}(\alpha)_i < -y_p \nabla l_{\text{mtl}}(\alpha)_p\}.\end{aligned} \quad (6.20)$$

To solve for the second index q in (6.20), the problem (6.18) with one variable p given needs to be solved. This is not difficult as we will show that analytical form of $\Delta l_{\text{mtl}}(\alpha)_{pq}^{\min}$ can be found. We have the following theorem similar to theorem 3 in (Fan,

Chen and Lin, 2005).

Theorem 5.1: If p,q is a violating pair from group r, let

$$a_{pq} = (\mathbf{z}_p \mathbf{g} \mathbf{z}_p) + (\mathbf{z}_q \mathbf{g} \mathbf{z}_q) - 2(\mathbf{z}_p \mathbf{g} \mathbf{z}_q) + \frac{1}{\gamma}((\mathbf{z}_p^r \mathbf{g} \mathbf{z}_p^r) + (\mathbf{z}_q^r \mathbf{g} \mathbf{z}_q^r) - 2(\mathbf{z}_p^r \mathbf{g} \mathbf{z}_q^r)) \quad \text{and}$$

$v_{pq} = -y_p \nabla l_{\text{mtl}}(\alpha_p) + y_q \nabla l_{\text{mtl}}(\alpha_q)$. If $a_{pq} > 0$, then (6.19) has the optimal objective value $-\frac{v_{pq}^2}{2a_{pq}}$ when $\Delta\alpha_p = \frac{y_p v_{pq}}{a_{pq}}$ and $\Delta\alpha_q = -\frac{y_q v_{pq}}{a_{pq}}$.

Proof: The proof is similar to theorem 3 in (Fan, Chen and Lin, 2005). Define $u_p = y_p \Delta\alpha_p$ and $u_q = y_q \Delta\alpha_q$. From the equality condition $y_p \Delta\alpha_p + y_q \Delta\alpha_q = 0$ in (6.19), we have $u_p = -u_q$ and

$$\Delta l_{\text{mtl}}(\alpha)_{pq} = \frac{1}{2} a_{pq} u_q^2 + v_{pq} u_q. \qquad (6.21)$$

If $a_{pq} > 0$, then (6.21) has the minimum at

$$u_q = -u_p = -\frac{v_{pq}}{a_{pq}} < 0, \qquad (6.22)$$

and the optimal objective value of (6.19) is $-\frac{v_{pq}^2}{2a_{pq}}$. Finally, we can show that value for u_p and u_q in (6.22) indeed satisfy the inequality conditions in (6.19). If $q \in I_{\text{low}}^r(\alpha)$, $\alpha_q = C$ implies that $y_q = 1$ and hence $\Delta\alpha_q = u_q < 0$, this satisfies the inequality condition in (6.19). Other cases are similar. Therefore u_p and u_q defined in (6.22) are optimal for (6.19). Therefore, optimal values for $\Delta\alpha_p$ and $\Delta\alpha_q$ can be derived: $\Delta\alpha_p = u_p y_p = \frac{y_p v_{pq}}{a_{pq}}$

and $\Delta\alpha_q = u_q y_q = -\frac{y_q v_{pq}}{a_{pq}}$.

Using Theorem 5.1, when $a_{pq} > 0$, the selection of second variable in (6.20) is reduced to a simple form:

$$q \in \arg\min_i \{-\frac{v_{pi}^2}{a_{pi}} \mid i \in I_{\text{low}}^r(\alpha), -y_i \nabla l_{\text{mtl}}(\alpha)_i < -y_p \nabla l_{\text{mtl}}(\alpha)_p\}. \quad (6.23)$$

However, if a_{pq} in Theorem 5.1 is non-positive, then variable selection in (6.23) will be invalid. This case is similar to the SMO for standard SVM where the kernel matrices are non-positive definite. It is proposed in (Fan, Chen and Lin, 2005) that an additional term can be added to the objective function so that the new optimization problem becomes convex. Therefore, we also add a similar term $\frac{\tau - a_{pq}}{4}(\Delta\alpha_p^2 + \Delta\alpha_q^2)$ to our objective function (6.19), where τ is a small positive number. The modified objective function $\Delta l_{\text{mtl}}(\alpha)_{pq}$ is:

$$\begin{aligned}\Delta l_{\text{mtl}}(\alpha)_{pq} &= \frac{1}{2}a_{pq}u_q^2 + v_{pq}u_q + \frac{\tau - a_{pq}}{4}(\Delta\alpha_p^2 + \Delta\alpha_q^2) \\ &= \frac{1}{2}a_{pq}u_q^2 + v_{pq}u_q + \frac{\tau - a_{pq}}{4}(u_q^2 + u_q^2) \\ &= \frac{1}{2}\tau u_q^2 + v_{pq}u_q.\end{aligned}$$

In this case, the minimum of the objective function can be obtained at

$$u_q = -u_p = -\frac{v_{pq}}{\tau} < 0, \quad (6.24)$$

and the minimum is $-\frac{v_{pq}^2}{2\tau}$. Similar to the proof in Theorem 5.1, the values of u_p and u_q

in (6.24) satisfy the inequality conditions of (6.19). Therefore, in this case the selection of second variable in (6.20) has the following simple form:

$$q \in \arg\min_i \{-\frac{v_{pi}^2}{\tau} \mid i \in I_{\text{low}}^r(\alpha), -y_i \nabla l_{\text{mtl}}(\alpha)_i < -y_p \nabla l_{\text{mtl}}(\alpha)_p\}. \quad (6.25)$$

B. Updating Lagrange Multipliers

The two variables α_p and α_q from same group r have already been selected as described in the previous subsection. Next, the two variables α_p and α_q need to be updated. The optimal values for $\Delta\alpha_p$ and $\Delta\alpha_q$ can be found in Theorem 5.1. Therefore, it is straightforward to update α_p and α_q as follows: $\alpha_p^{\text{new}} = \alpha_p + \Delta\alpha_p$ and $\alpha_q^{\text{new}} = \alpha_q + \Delta\alpha_q$. The only problem is that if α_p^{new} or α_q^{new} falls out of range $[0, C]$, they need to be clipped to be in the range $[0, C]$. Therefore, given a_{pq} and v_{pq} defined in Theorem 5.1, the Lagrange multipliers α_p and α_q are first updated as follows:

$$\begin{aligned} \alpha_p^{\text{new}} &= \begin{cases} \alpha_p + y_p v_{pq} / a_{pq} & \text{if } a_{pq} > 0; \\ \alpha_p + y_p v_{pq} / \tau & \text{if } a_{pq} \le 0. \end{cases} \\ \alpha_q^{\text{new}} &= \begin{cases} \alpha_q - y_q v_{pq} / a_{pq} & \text{if } a_{pq} > 0; \\ \alpha_q + y_q v_{pq} / \tau & \text{if } a_{pq} \le 0. \end{cases} \end{aligned} \quad (6.26)$$

Then, α_p^{new} and α_q^{new} need to be clipped so that they are in the range $[0, C]$. The Lagrange multiplier α_p^{new} is first clipped as follows:

$$\alpha_p^{\text{new,clipped}} = \begin{cases} 0 & \text{if } \alpha_p^{\text{new}} < 0; \\ \text{C} & \text{if } \alpha_p^{\text{new}} > C; \\ \alpha_p^{\text{new}} & \text{Otherwise.} \end{cases} \quad (6.27)$$

In order to satisfy the equality condition (6.11) the second Lagrange multiplier α_q^{new} needs to altered to reflect the updated $\alpha_p^{\text{new,clipped}}$:

$$\alpha_q^{\text{new}} = y_q(y_p\alpha_p + y_p\alpha_p - y_p\alpha_p^{\text{new,clipped}}).$$

Finally, α_q^{new} is clipped to satisfy the constraint $[0, C]$:

$$\alpha_q^{\text{new,clipped}} = \begin{cases} 0 & \text{if } \alpha_q^{\text{new}} < 0; \\ \text{C} & \text{if } \alpha_q^{\text{new}} > C; \\ \alpha_q^{\text{new}} & \text{Otherwise.} \end{cases} \qquad (6.28)$$

C. Computing the threshold

Estimation of t classification functions $f_r(\mathbf{x}), r = 1,\text{K}, t$ requires calculation of the thresholds $b + d_r$. Only a few α_i in each group r will be greater than zero. The corresponding $\mathbf{x}_i$ are the support vectors, which lie on the margin borders and satisfy $y_i((\mathbf{w} \bullet \mathbf{z}_i) + b + (\mathbf{w}_r \bullet \mathbf{z}_i^r) + d_r) = 1$. Since $y_i = \pm 1$, these support vectors also satisfy $b + d_r = y_i - (\mathbf{w} \bullet \mathbf{z}_i) - (\mathbf{w}_r \bullet \mathbf{z}_i^r)$. Suppose there are N_{sv}^r support vectors in group r. In practice, it is more robust to calculate the bias term by averaging over all N_{sv}^r support vectors:

$$b + d_r = \frac{1}{N_{\text{sv}}^r} \sum_{i \in T_r, \alpha_i > 0} y_i - (\mathbf{w} \bullet \mathbf{z}_i) - (\mathbf{w}_r \bullet \mathbf{z}_i^r).$$

6.4 GSMO for SVM+MTL Regression

The SMO training algorithm of SVM regression is different from that of SVM

classification as the dual formulation of SVM regression has one more variable (Shevade et al. 2000). Similarly, the dual of SVM+MTL regression, which is given next, has one more variable than dual of SVM+MTL classification (6.10).

GSMO for SVM+MTL regression needs to solve the following dual formulation:

$$\begin{aligned}\min_{\alpha,\alpha^*}\ & \varepsilon\sum_{i=1}^{n}(\alpha_i^*+\alpha_i)-\sum_{i=1}^{n}(\alpha_i^*-\alpha_i)y_i+\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)(\mathbf{z}_i\mathbf{g}\mathbf{z}_j)\\ &+\frac{1}{2\gamma}\sum_{r=1}^{t}\sum_{i,j\in T_r}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)(\mathbf{z}_i^r\mathbf{g}\mathbf{z}_j^r)\\ \text{subject to: }& \sum_{i\in T_r}(\alpha_i^*-\alpha_i)=0,\ r=1,\text{K},t\\ & 0\le\alpha_i\le C,\ 0\le\alpha_i^*\le C,\ i=1,\text{K},n\end{aligned} \tag{6.29}$$

In order to reduce the number of variables in (6.29) to 1, let $\lambda_i=\alpha_i^*-\alpha_i$, the new optimization becomes:

$$\begin{aligned}\min_{\alpha,\alpha^*}\ & \varepsilon\sum_{i=1}^{n}|\lambda_i|-\sum_{i=1}^{n}\lambda_i y_i+\frac{1}{2}\sum_{i,j=1}^{n}\lambda_i\lambda_j(\mathbf{z}_i\mathbf{g}\mathbf{z}_j)\\ &+\frac{1}{2\gamma}\sum_{r=1}^{t}\sum_{i,j\in T_r}\lambda_i\lambda_j(\mathbf{z}_i^r\mathbf{g}\mathbf{z}_j^r)\\ \text{subject to: }& \sum_{i\in T_r}\lambda_i=0,\ r=1,\text{K},t\\ & -C\le\lambda_i\le C,\ i=1,\text{K},n\end{aligned} \tag{6.30}$$

and t estimated regression functions are:

$$f_r(\mathbf{x})=\sum_{i=1}^{n}\lambda_i(\mathbf{z}_i\mathbf{g}\mathbf{z})+b+\frac{1}{\gamma}\sum_{i\in T_r}\lambda_i(\mathbf{z}_i^r\mathbf{g}\mathbf{z}^r)+d_r, r=1,\text{K},t.$$

A. Solving for Two Lagrange Multipliers

The objective function of (6.30) is denoted as $l(\lambda)$. In order to minimize $l(\lambda)$, we choose two variables denoted as λ_p and λ_q from the same group r. The objective function $l(\lambda)$

is then optimized over these two variables. During this optimization step, the remaining variables are fixed as constants. The objective function $l(\lambda)$ can be written as a function of variables λ_p and λ_q:

$$\begin{aligned} l(\lambda_p, \lambda_q) = \varepsilon |\lambda_p| + \varepsilon |\lambda_q| - \lambda_p y_p - \lambda_q y_q + \frac{1}{2}\lambda_p^2 (\mathbf{z}_p \mathbf{z}_p) \\ + \frac{1}{2}\lambda_q^2 (\mathbf{z}_q \mathbf{z}_q) + \lambda_p \lambda_q (\mathbf{z}_p \mathbf{z}_q) + \frac{1}{2\gamma}\lambda_p^2 (\mathbf{z}_p^r \mathbf{z}_p^r) \\ + \frac{1}{2\gamma}\lambda_q^2 (\mathbf{z}_q^r \mathbf{z}_q^r) + \frac{1}{\gamma}\lambda_p \lambda_q (\mathbf{z}_p^r \mathbf{z}_q^r) + \lambda_p M_p + \lambda_q M_q + C_1 \end{aligned}$$

where $M_i = \sum_{j \neq p,q} \lambda_j (\mathbf{z}_i \mathbf{z}_j) + \frac{1}{\gamma} \sum_{j \in T_r, j \neq p,q} \lambda_j (\mathbf{z}_i^r \mathbf{z}_j^r)$ and C_1 is constant. The value $M_i, i = p, q$ can be computed with old parameters. Since we have the linear constraint $\sum_{i \in T_r} \lambda_i = 0$ and variables other than λ_p and λ_q are fixed during the optimization step. Then the sum of these two variables $s = \lambda_p + \lambda_q$ will remain unchanged before and after the optimization.

In order to minimize the objective function $l(\lambda_p, \lambda_q)$ with respect to λ_p and λ_q, λ_p is replaced with $s - \lambda_q$. Then $l(\lambda_p, \lambda_q)$ becomes

$$\begin{aligned} l(\lambda_q) = \varepsilon |s - \lambda_q| + \varepsilon |\lambda_q| - (s - \lambda_q) y_p - \lambda_q y_q + \frac{1}{2}(s - \lambda_q)^2 (\mathbf{z}_p \mathbf{z}_p) \\ + \frac{1}{2}\lambda_q^2 (\mathbf{z}_q \mathbf{z}_q) + (s - \lambda_q)\lambda_q (\mathbf{z}_p \mathbf{z}_q) + \frac{1}{2\gamma}(s - \lambda_q)^2 (\mathbf{z}_p^r \mathbf{z}_p^r) \\ + \frac{1}{2\gamma}\lambda_q^2 (\mathbf{z}_q^r \mathbf{z}_q^r) + \frac{1}{\gamma}(s - \lambda_q)\lambda_q (\mathbf{z}_p^r \mathbf{z}_q^r) + (s - \lambda_q) M_p + \lambda_q M_q + C_1 \end{aligned}$$

The two variables λ_p and λ_q must fulfill the constraints in (6.30), therefore, the point with coordinates λ_p and λ_q must lie on the diagonal line in a box (Figure 6.1). The

following bounds apply to λ_q: $L = \max(s - C, -C), H = \min(C, s + C)$.

To minimize $l(\lambda_q)$, we need to compute its derivative with respect to λ_q:

$$\begin{aligned}\frac{\partial l(\lambda_q)}{\partial \lambda_q} &= \varepsilon(\mathrm{sgn}(\lambda_q) - \mathrm{sgn}(s - \lambda_q)) + y_p - y_q + (\lambda_q - s)(\mathbf{z}_p \mathbf{z}_p) \\ &+ \lambda_q(\mathbf{z}_q \mathbf{z}_q) + (s - 2\lambda_q)(\mathbf{z}_p \mathbf{z}_q) + \frac{1}{\gamma}(\lambda_q - s)(\mathbf{z}_p^r \mathbf{z}_p^r) \\ &+ \frac{1}{\gamma}\lambda_q(\mathbf{z}_q^r \mathbf{z}_q^r) + \frac{1}{\gamma}(s - 2\lambda_q)(\mathbf{z}_p^r \mathbf{z}_q^r) - M_p + M_q\end{aligned} \tag{6.31}$$

Setting equation (6.31) to zero yields:

$$\begin{aligned}&\lambda_q^{\mathrm{new}}((\mathbf{z}_p \mathbf{z}_p) + (\mathbf{z}_q \mathbf{z}_q) - 2(\mathbf{z}_p \mathbf{z}_q) + \frac{1}{\gamma}((\mathbf{z}_p^r \mathbf{z}_p^r) + (\mathbf{z}_q^r \mathbf{z}_q^r) - 2(\mathbf{z}_p^r \mathbf{z}_q^r))) \\ &= y_q - y_p + s((\mathbf{z}_p \mathbf{z}_p) - (\mathbf{z}_p \mathbf{z}_q) + \frac{1}{\gamma}((\mathbf{z}_p^r \mathbf{z}_p^r) - (\mathbf{z}_p^r \mathbf{z}_q^r))) \\ &+ \varepsilon(\mathrm{sgn}(s - \lambda_q^{\mathrm{new}}) - \mathrm{sgn}(\lambda_q^{\mathrm{new}})) + M_p - M_q \\ &= y_q - y_p + (\lambda_p + \lambda_q)((\mathbf{z}_p \mathbf{z}_p) - (\mathbf{z}_p \mathbf{z}_q) + \frac{1}{\gamma}((\mathbf{z}_p^r \mathbf{z}_p^r) - (\mathbf{z}_p^r \mathbf{z}_q^r))) \\ &+ \varepsilon(\mathrm{sgn}(s - \lambda_q^{\mathrm{new}}) - \mathrm{sgn}(\lambda_q^{\mathrm{new}})) + f_r(\mathbf{x}_p) - \lambda_p(\mathbf{z}_p \mathbf{z}_p) - \lambda_q(\mathbf{z}_p \mathbf{z}_q) \\ &- b - \frac{1}{\gamma}(\lambda_p(\mathbf{z}_p^r \mathbf{z}_p^r) + \lambda_q(\mathbf{z}_p^r \mathbf{z}_q^r)) - d_r - (f_r(\mathbf{x}_q) - \lambda_p(\mathbf{z}_p \mathbf{z}_q) \\ &- \lambda_q(\mathbf{z}_q \mathbf{z}_q) - b - \frac{1}{\gamma}(\lambda_p(\mathbf{z}_p^r \mathbf{z}_q^r) + \lambda_q(\mathbf{z}_q^r \mathbf{z}_q^r)) - d_r) \\ &= f_r(\mathbf{x}_p) - y_p - (f_r(\mathbf{x}_q) - y_q) + \varepsilon(\mathrm{sgn}(\lambda_p^{\mathrm{new}}) - \mathrm{sgn}(\lambda_q^{\mathrm{new}})) \\ &+ \lambda_q((\mathbf{z}_p \mathbf{z}_p) + (\mathbf{z}_q \mathbf{z}_q) - 2(\mathbf{z}_p \mathbf{z}_q) + \frac{1}{\gamma}((\mathbf{z}_p^r \mathbf{z}_p^r) + (\mathbf{z}_q^r \mathbf{z}_q^r) - 2(\mathbf{z}_p^r \mathbf{z}_q^r))) \\ &= \lambda_q((\mathbf{z}_p \mathbf{z}_p) + (\mathbf{z}_q \mathbf{z}_q) - 2(\mathbf{z}_p \mathbf{z}_q) + \frac{1}{\gamma}((\mathbf{z}_p^r \mathbf{z}_p^r) + (\mathbf{z}_q^r \mathbf{z}_q^r) - 2(\mathbf{z}_p^r \mathbf{z}_q^r))) \\ &+ E_p - E_q + \varepsilon(\mathrm{sgn}(\lambda_p^{\mathrm{new}}) - \mathrm{sgn}(\lambda_q^{\mathrm{new}})),\end{aligned} \tag{6.32}$$

Where $E_i = f_r(\mathbf{x}_i) - y_i, i = p, q$ is the estimation error using old parameter values. From equation (6.32), a recursive update rule for λ_q can be written as:

$$\lambda_q^{\text{new}} = \lambda_q + \frac{1}{\eta}(E_p - E_q + \varepsilon(\text{sgn}(\lambda_p^{\text{new}}) - \text{sgn}(\lambda_q^{\text{new}}))),$$

Where $\eta = (\mathbf{z}_p \mathbf{z}_p) + (\mathbf{z}_q \mathbf{z}_q) - 2(\mathbf{z}_p \mathbf{z}_q) + \frac{1}{\gamma}((\mathbf{z}_p^r \mathbf{z}_p^r) + (\mathbf{z}_q^r \mathbf{z}_q^r) - 2(\mathbf{z}_p^r \mathbf{z}_q^r))$. As a next step, the constrained minimum is found by clipping the unconstrained minimum to the ends of the line segment:

$$\lambda_q^{\text{new,clipped}} = \begin{cases} H & \text{if} \quad \lambda_q^{\text{new}} \geq H; \\ \lambda_q^{\text{new}} & \text{if} \ L \leq \lambda_q^{\text{new}} \leq H; \\ L & \text{if} \quad \lambda_q^{\text{new}} \leq L. \end{cases}$$

The value of λ_p^{new} is computed from the new clipped λ_q: $\lambda_p^{\text{new}} = s - \lambda_q^{\text{new,clipped}}$.

Finally, the pseudo code for the analytical step for GSMO for SVM+MTL regression can be found in algorithm 6.2.

B. Heuristics for Choosing Two Variables

Similar to Platt's SMO, our generalized SMO uses heuristics for selecting two variables subject to joint optimization. The first variable is chosen by an outer loop which iterates over the entire training set to determine whether each example violates a Karush-Kuhn-Tucker (KKT) condition. The KKT conditions for SVM+MTL regression are:

$$\begin{array}{lll} \lambda_i = 0 & \Leftrightarrow & |\, y_i - f_r(\mathbf{x}_i) \,| \leq \varepsilon \\ 0 < \lambda_i < \text{C} & \Leftrightarrow & |\, y_i - f_r(\mathbf{x}_i) \,| = \varepsilon \quad i \in T_r \\ \lambda_i = C & \Leftrightarrow & |\, y_i - f_r(\mathbf{x}_i) \,| \geq \varepsilon \end{array}$$

The global minimum has been reached when no parameter violates any KKT condition.

Algorithm 6.2 Analytical step for GSMO for SVM+MTL regression

1. $s = \lambda_p + \lambda_q$
2. $\eta = (\mathbf{z}_p \mathbf{g} \mathbf{z}_p) + (\mathbf{z}_q \mathbf{g} \mathbf{z}_q) - 2(\mathbf{z}_p \mathbf{g} \mathbf{z}_q) + \frac{1}{\gamma}((\mathbf{z}_p^r \mathbf{g} \mathbf{z}_p^r) + (\mathbf{z}_q^r \mathbf{g} \mathbf{z}_q^r) - 2(\mathbf{z}_p^r \mathbf{g} \mathbf{z}_q^r))$
3. $\Delta = \frac{2\varepsilon}{\eta}$
4. $\lambda_q^{\text{new}} = \lambda_q + \frac{1}{\eta}(E_p - E_q)$
5. $\lambda_p^{\text{new}} = s - \lambda_q^{\text{new}}$
6. If $\lambda_p^{\text{new}} \lambda_p^{\text{new}} < 0$ then
7. If $|\lambda_p^{\text{new}}| \geq \Delta \wedge |\lambda_q^{\text{new}}| \geq \Delta$ then
8. $\lambda_q^{\text{new}} = \lambda_q^{\text{new}} - \text{sgn}(\lambda_q^{\text{new}})\Delta$
9. Else
10. $\lambda_q^{\text{new}} = \text{step}(|\lambda_q^{\text{new}}| - |\lambda_p^{\text{new}}|)s$
11. End if
12. End if
13. $L = \max(s - C, -C)$
14. $H = \min(C, s + C)$
15. $\lambda_q^{\text{new,clipped}} = \min(\max(\lambda_q^{\text{new}}, L), H)$
16. $\lambda_p^{\text{new}} = s - \lambda_q^{\text{new,clipped}}$

The outer loop first searches over the entire training set, after one loop, it will search over the samples that are not on bound. The outer loop keeps alternating between passes over entire training set and passes over the non-bound subset.

The GSMO then needs to find the second variable λ_q. It first finds a non-bound variable λ_q where $|E_p - E_q|$ is maximum. If the joint optimization of λ_p and λ_q is not successful. GSMO will search over all the non-bound variables. If none of these variables work, GSMO will finally search the second variable λ_q over the entire training set.

C. Updating the Threshold

Similar to SVM, we calculate two candidate updates to update the threshold $b + d_r$ for SVM+MTL. The first update, if used along with the new parameters, forces the SVM+MTL to have $f_r(\mathbf{x}_p) = y_p$. The second forces $f_r(\mathbf{x}_q) = y_q$. If neither update for the other two parameters hits a constraint, then the two candidate updates will be identical. Otherwise, we average the candidate updates.

$$b^{p,\text{new}} + d_r^{p,\text{new}} = b^p + d_r^p - E_p - (\lambda_p^{\text{new}} - \lambda_p)(\mathbf{z}_p \mathbf{z}_p) - (\lambda_q^{\text{new,clipped}} - \lambda_q)(\mathbf{z}_p \mathbf{z}_q) - \frac{1}{\gamma}((\lambda_p^{\text{new}} - \lambda_p)(\mathbf{z}_p^r \mathbf{z}_p^r) + (\lambda_q^{\text{new,clipped}} - \lambda_q)(\mathbf{z}_p^r \mathbf{z}_q^r))$$

$$b^{q,\text{new}} + d_r^{q,\text{new}} = b^q + d_r^q - E_q - (\lambda_p^{\text{new}} - \lambda_p)(\mathbf{z}_p \mathbf{z}_q) - (\lambda_q^{\text{new,clipped}} - \lambda_q)(\mathbf{z}_q \mathbf{z}_q) - \frac{1}{\gamma}((\lambda_p^{\text{new}} - \lambda_p)(\mathbf{z}_p^r \mathbf{z}_q^r) + (\lambda_q^{\text{new,clipped}} - \lambda_q)(\mathbf{z}_q^r \mathbf{z}_q^r))$$

6.5 Empirical Comparisons

This section compares computational efficiency of the proposed GSMO algorithms for SVM+MTL with a general-purpose quadratic programming solver based on CVX optimization package (Grant and Boyd, 2010). Comparisons use two synthetic datasets and a real dataset. The synthetic dataset for classification is generated as follows:

(1) Let number of input features be $d = 30$, and number of tasks(groups) be $t = 3$.

(2) Generate $\mathbf{x} \in R^{30}$ with each component $x_i \sim \textit{uniform}(0,1), i = 1,...,30$.

(3) The coefficient vectors of three (linear) regression tasks are specified as:

$$\beta_1 = [1,1,2,3,3,1,1,1,1,0,2,0,2,2,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0],$$
$$\beta_2 = [1,1,2,3,3,1,1,1,1,0,2,0,2,2,0,0,0,0,2,0,2,0,0,0,0,0,0,0,0,0],$$
$$\beta_3 = [1,1,2,3,3,1,1,1,0,1,0,0,3,0,0,2,0,2,0,2,0,0,0,0,0,0,0,0,0,0].$$

(4) For each task and each data vector, $y = \text{sign}(\beta_i \mathbf{x} + n_{\text{noise}})$, where n_{noise} is Gaussian noise with $\sigma_{noise} = 0.5$.

Three different data sets are generated for each experiment. Training data is used for model estimation, validation data used for model selection (parameter tuning) and test data for evaluating generalization performance. Both validation and test sets have 3000 samples, 1000 for each group. Training set size is varied from 300 to 2400 to test our algorithms on different sample size. We used linear kernel for decision space and RBF kernel for correcting space. Thus, in total, we have 3 tuning parameters for SVM+MTL: C, γ, σ. The possible choices for tuning parameters are $C = [0.1, 1, 10]$, $\gamma = [0.1, 1, 10]$, $\sigma = [0.5, 1, 2]$. Comparisons are shown for different number of training samples. For each training size, an SVM+MTL classifier is estimated using GSMO and CVX algorithms. The total training time, averaged over 10 trials, are recorded for both algorithms. We ran the experiments on a computer with core 2 duo 3G CPU and 4G memory. The results shown in Table 6.1 indicate that GSMO achieves significant speed-up. For large-size data sets (over 3K), the CVX optimization package crashes (runs out of memory). In all cases, GSMO and CVX implementations of SVM+MTL training yield the same (or very close) generalization performance. Since our implementations use different programming languages (C for GSMO and Matlab for CVX), it may be better to present comparison using complexity analysis. The complexity of training method by CVX is $O(n^3)$ as $(16698/3627) = (1500/900)^3$ and $(67111/16698) = (2400/1500)^3$ while the complexity

of GSMO is around $O(n^{2.2})$ as $(23.41/8.64) = (1500/900)^2$ and $(66.63/23.41) = (2400/1500)^{2.2}$.

We also use performed comparisons for a real classification dataset corresponding to radar-based sensing of landmines (Xue et al., 2007). The dataset has 29 tasks, where each task corresponds to data collected from a unique landmine field. Each data sample is represented by a 9-dimensional feature vector and the corresponding binary label (1 for landmine and 0 for clutter). Data from 6 different tasks are selected and the number of samples in each task is varied from 40 to 160. For each task, one third of training samples correspond to landmines and two thirds to clutter. Five-fold cross-validation is used to estimate generalization performance. We follow application of resampling under MTL setting as described in chapter 3,4. Comparison results are shown in Table 6.2. Similar to earlier results using synthetic data, computational complexity of CVX method scales as $O(n^3)$ i.e. $(590/80) = (80/40)^{2.9}$ and $(4596/590) = (160/80)^3$. In contrast, the complexity of GSMO is around $O(n^{2.2})$ i.e. $(2.32/0.49) = (80/40)^{2.2}$ and $(9.85/2.32) = (160/80)^{2.1}$.

The synthetic data for regression is described in Chapter 5. The dimension of the data is 30, the training size N is 50 for each group and the variance of noise $\sigma_{\text{noise}} = 1$. We choose linear kernel for common space and RBF kernel for unique correcting space. The experimental results are in Table 6.3-6.5. We can see that GSMO is much faster while the prediction error is similar.

6.6 Conclusion

This paper generalizes Platt's SMO for SVM+ based Multi-Task Learning formulation known as SVM+MTL (Liang and Cherkassky, 2008; Cai and Cherkassky, 2009). The proposed Generalized SMO also breaks the large QP problem during SVM+MTL training into a series of smallest possible QP sub-problems, which can be solved analytically. Proposed GSMO algorithm, reflects interaction between samples from different groups. We also presented empirical comparisons between the proposed GSMO and a general-purpose QP solver based on CVX optimization package. These results indicate that the GSMO algorithm is two orders of magnitude faster, and can handle much larger training sets, than existing implementations based on CVX optimization package (Liang and Cherkassky, 2008; Cai and Cherkassky, 2009).

Due to similarity between our GSMO and the original Platt's SMO for standard SVM, the proposed GSMO can handle as many samples as well-known SVM software packages, such as Libsvm (Fan, Chen and Lin, 2005).

Table 6.1 Comparison between GSMO and CVX, synthetic data for classification

Training Size	GSMO error (%)	CVX error (%)	GSMO time (s)	CVX time (s)
900	10.0	9.8	8.64	3627
1500	8.6	9.0	23.41	16698
2400	7.0	7.0	66.63	67111

Table 6.2 Comparison between GSMO and CVX, Landmine data

Size per group	GSMO error (%)	CVX error (%)	GSMO time (s)	CVX time (s)
40	15	15.4	0.49	80
80	9.6	9.6	2.32	590
160	9.5	9.4	9.85	4596

Table 6.3 Comparison between GSMO and CVX, synthetic data for regression. Results are averaged over 10 trials

$(C = 1, \gamma = 0.1, \sigma = 5, \varepsilon = 0.0001)$

Training Algorithm	CPU time	Prediction error (%)
GSMO	1.068	1.853
CVX	5.479	1.857

Table 6.4 Comparison between GSMO and CVX, synthetic data for regression. Results are averaged over 10 trials

$(C = 1, \gamma = 0.1, \sigma = 1, \varepsilon = 0.0001)$

Training Algorithm	CPU time	Prediction error (%)
GSMO	0.542	2.146
CVX	5.068	2.146

Table 6.5 Comparison between GSMO and CVX, synthetic data for regression. Results are averaged over 10 trials

$(C = 1, \gamma = 1, \sigma = 5, \varepsilon = 0.0001)$

Training Algorithm	CPU time	Prediction error (%)
GSMO	0.354	2.290
CVX	5.448	2.288

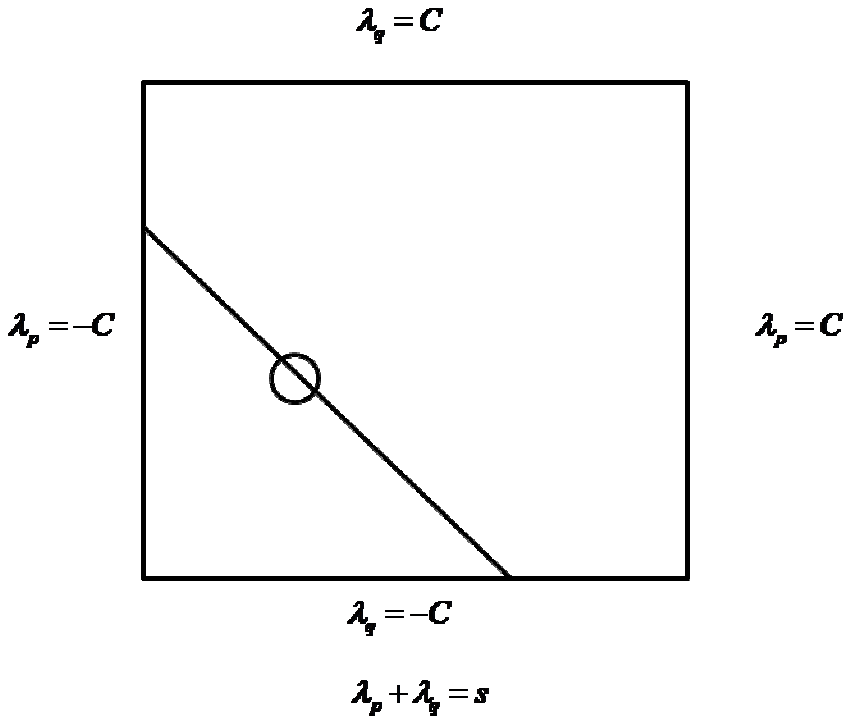

Figure 6.1 The two variables λ_p and λ_q must fulfill the constraints in (6.30). The equality constraint causes them to lie on a diagonal line while the inequality constraints cause them to lie in the box.

Chapter 7 Summary and My Contributions

There are four major contributions described in this dissertation:

First, we introduced a SVM+ based approach to feature selection. We show that features (not available during test) removed by feature selection can be used by SVM+ as privileged information to help improve generalization.

Second, we compared SVM+MTL classification with SVM, SVM+ and rMTL. We have also incorporated unequal misclassification cost into SVM+MTL classification.

Third, we proposed SVM+MTL regression algorithm. Empirical results show that SVM+MTL regression outperforms SVM and SVM+ when data can be naturally partitioned into several groups.

Fourth, we designed a generalized SMO algorithm to train SVM+MTL. Empirical results show that our GSMO performs much faster than existing implementations based on CVX optimization package.

There are still many open and challenging research questions. Many existing learning approaches assume a standard inductive learning formulation, where the goal is to estimate a predictive model from finite training data. While this inductive setting is

very general and commonly accepted, it can not be taken for granted. As pointed out in (Cherkassky and Mulier, 2007), *future progress in predictive learning is likely to occur due to better understanding (and acceptance) of non standard learning inference, rather than marginal improvements of learning algorithms implementing standard inductive inference*. This line of thinking is in complete agreement with the work shown in this thesis, where we have demonstrated the advantages of methodologies based on new learning formulations compared to the methods based on standard inductive learning.

Publications from this thesis

Vladimir Cherkassky, Feng Cai and Lichen Liang, Predictive learning with sparse heterogeneous data, *International Joint Conference on Neural Networks (IJCNN)*, 2009.

Feng Cai and Vladimir Cherkassky, SVM+ regression and multi-task learning, *International Joint Conference on Neural Networks (IJCNN)*, 2009.

Lichen Liang, Feng Cai and Vladimir Cherkassky, Predictive learning with structured (grouped) data, *Neural Networks*, 22: 766 – 773, 2009.

Feng Cai, Vladimir Cherkassky, Daniel Weisdorf, Mukta Arora, Brian Van Ness and Bharat Thyagarajan, Predictive modeling of transplant-related mortality, *Design of Medical Devices Conference*, 2010

Feng Cai and Vladimir Cherkassky, Generalized SMO algorithm for SVM based multi-task learning, *IEEE Trans. Neural Networks*, under second round review.

Feng Cai, Han-Tai Shiao and Vladimir Cherkassky, SVM+ Based Multi-Task Learning, to be submitted.

Feng Cai and Vladimir Cherkassky, SVM+ Based Feature Selection, to be submitted.

References

V. N. Vapnik, *Estimation of dependences based on empirical data*, Springer Verlag, New York, 1982.

V. N. Vapnik, *The nature of statistical learning theory*, Springer, New York, 1995.

V. N. Vapnik, *Statistical learning theory*, Springer, New York, 1998.

V. N. Vapnik, *Empirical inference science: afterword of 2006*, Springer, New York, 2006.

V. Cherkassky and F. Mulier, *Learning from data*, John Wiley & Sons, New York, 1998.

V. Cherkassky and F. Mulier, *Learning from data*, John Wiley & Sons, New York, second edition, 2007.

V. Vapnik and A. Vashist, A new learning paradigm: learning using privileged information, *Neural Networks*, 22(5-6): 544 – 557, 2009.

V. Vapnik, A. Vashist and N. Pavlovitch, learning using hidden information (learning with teacher), *IJCNN*, 3188 – 3195, 2009.

T. Hastie, R. Tibshirani and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*, Springer Verlag, New York, 2001.

L. Liang and V. Cherkassky, Connection between SVM+ and multi-task learning, *IJCNN*, Hong Kong, 2008.

L. Liang, Application and development of new learning methodologies for fMRI data analysis, PhD thesis, University of Minnesota, 2008.

T. Evgeniou and M. Pontil, Regularized multi-task learning, *In proc. 17th SIGKDD conf. on Knowledge Discovery and Data Mining*, 2004.

D. Michie, D. J. Spiegelhalter and C. C. Taylor, Machine learning, neural and statistical classification, Prentice Hall, 1994.

I. W. Tsang, J. T. Kwok and P. Cheung, Core vector machines: fast SVM training on very large data sets, *Journal of Machine Learning Research*, 6: 363 – 392, 2005.

J. Platt, Sequential minimal optimization: a fast algorithm for training support vector machines, *Technical Report MSR-TR-98-14*, Microsoft Research, 1998.

G. Obozinski, B. Taskar and M. Jordan, Multi-task feature selection, Technical Report, Department of Statistics, University of California, Berkeley, 2006.

A. Argyriou, T. Evgeniou and M. Pontil, Multi-task feature learning, *NIPS*, 2006.

B. Bakker and T. Heskes, Task clustering and gating for Bayesian multi-task learning, *Journal of Machine Learning Research*, 4: 83-89, 2003.

R. Ando and T. Zhang, A framework for learning predictive structures from multiple tasks and unlabeled data, *Journal of Machine Learning Research*, 6: 1817-1853, 2005.

X. Liao and L. Carin, Radial basis function network for multi-task learning, *NIPS*, 2005.

R. Raina, A. Y. Ng and D. Koller, Constructing informative priors using transfer learning, *ICML*, 2006.

N. D. Lawrence and J. Platt, Learning to learn with the informative vector machine, *ICML*, 2004.

Y. Xue, X. Liao, L. Carin and B. Krishnapuram, Multi-task learning for classification with Dirichlet process priors, *Journal of Machne Learning Research*, 8: 35-63, 2007.

V. Cherkassky and Y. Ma, Practical selection of svm parameters and noise estimation for svm regression, Neural Networks, 17: 113-126, 2004.

I. Guyon and A. Ellisseeff, An introduction to variable and feature selection, *Journal of Machine Learning* Research, 3: 1157-1182, 2003.

G. H. John, R. Kohavi and P. Pfleger, Irrelevant features and the subset selection problem, *Machine Learning: Proceedings of the Eleventh Internation Conference*, 1994.

M. Grant and S. Boyd, *CVX: Matlab software for disciplined convex programming*, version 1.21, http://cvxr.com/cvx, Dec. 2010.

S. S. Keerthi, S. K. Shevade, C. Bhattacharyya and K. R. K. Murphy, Improvements to Platt's SMO algorithm for svm classifier design, *Technical Report CD-99-14*, National University of Singapore, 1999.

R. E. Fan, P. H. Chen and C. J. Lin, Working set selection using second order information for training SVM, Journal of Machine Learning Research, 6: 243-264, 2005.

G. W. Flake and S. R. Lawrence, Efficient SVM regression training with SMO, Machine Learning, 46(1-3): 271 – 290, 2002.

S. K. Shevade, S. S. Keerthi, C. Bhattacharyya and K. R. K. Murphy, Improvements to the SMO algorithm for SVM regression, IEEE Tran. Neural Networks, 11(5): 1188-1193, 2000.

F. Fleuret, Fast binary feature selection with conditional mutual information, *Journal of Machine Learning Research*, 5: 1531 – 1555, 2004.

R. Fisher, The use of multiple measurements in Taxonomic problems, *Annals of Eugenics*, 7, p. 179-188, 1995.

G. Kitagawa, Monte-Carlo filter and smoother for non-Gaussian nonlinear state space models, J. Comput. Graph. Statist., 1:1-25, 1996

I. Guyon and S. Gunn, NIPS feature selection challenge, 2003.