Local Learning vs. Global Learning: An Introduction to Maxi-Min Margin Machine

K. Huang, H. Yang, I. King, and M.R. Lyu

Department of Computer Science and Engineering,
The Chinese University of Hong Kong,
Shatin, N.T., Hong Kong
{kzhuang, hqyang, king, lyu}@cse.cuhk.edu.hk

Abstract. We present a unifying theory of the Maxi-Min Margin Machine (M^4) that subsumes the Support Vector Machine (SVM), the Minimax Probability Machine (MPM), and the Linear Discriminant Analysis (LDA). As a unified approach, M^4 combines some merits from these three models. While LDA and MPM focus on building the decision plane using global information and SVM focuses on constructing the decision plane in a local manner, M^4 incorporates these two seemingly different yet complementary characteristics in an integrative framework that achieves good classification accuracy. We give some historical perspectives on the three models leading up to the development of M^4. We then outline the M^4 framework and perform investigations on various aspects including the mathematical definition, the geometrical interpretation, the time complexity, and its relationship with other existing models.

Key words: Classification, Local Learning, Global Learning, Hybrid Learning, M^4, Unified Framework

1 Introduction

When constructing a classifier, there is a dichotomy in choosing whether to use local vs. global characteristics of the input data. The framework of using global characteristics of the data, which we refer to as global learning, enjoys a long and distinguished history. When studying real-world phenomena, scientists try to discover the fundamental laws or underlying mathematics that govern these complex phenomena. Furthermore, in practice, due to incomplete information, these phenomena are usually described by using probabilistic or statistical models on sampled data. A common methodology found in these models is to fit a density on the observed data. With the learned density, people can easily perform prediction, inference, and marginalization tasks.

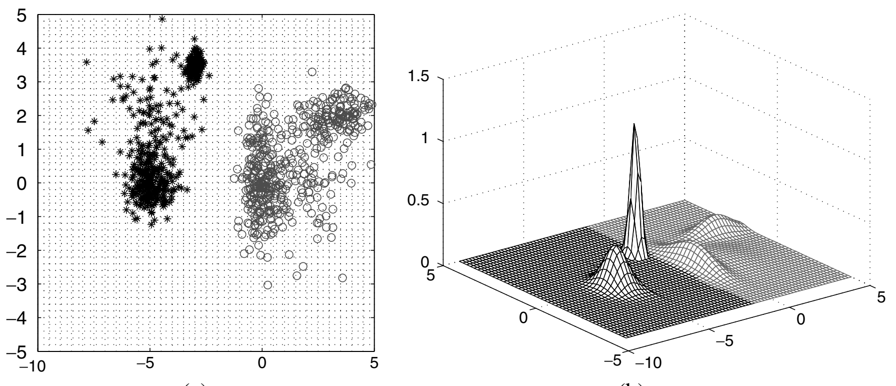


Fig. 1. An illustration of distribution-based classification (also known as the Bayes optimal decision theory). Two Gaussian mixtures are engaged to model the distribution of the two classes of data respectively. The distribution can then be used to construct the decision plane

One type of global learning is generative learning. By assuming a specific model on the observed data, e.g., a Gaussian distribution or a mixture of Gaussian, the phenomena can therefore be described or re-generated. Figure 1(a) illustrates such an example. In this figure, two classes of data are plotted as $*$'s for the first class and $\circ$'s for the other class. The data can thus be modelled as two different mixtures of Gaussian distributions. By knowing only the parameters of these distributions, one can then summarize the phenomena. Furthermore, as illustrated in Figure 1(b), one can clearly employ learned densities to distinguish one class of data from the other class (or simply know how to separate these two classes). This is the well-known Bayes optimal decision problem [1, 2].

One of the main difficulties found in global learning methodologies is the *model selection* problem. More precisely one still needs to select a suitable and appropriate model and its parameters in order to represent the observed data. This is still an open and on-going research topic. Some researchers have argued that it is difficult if not impossible to obtain a general and accurate global learning. Hence, local learning has recently attracted much interests.

Local learning [3, 4, 5] focuses on capturing only useful local information from the observed data. Furthermore, recent research progress and empirical studies demonstrate that the local learning paradigm is superior to global learning in many classification domains.

Local learning is more task-oriented since it omits an intermediate density modelling step in classification tasks. It does not aim to estimate a density from data as in global learning. In fact, it even does not intend to build an accurate model to fit the observed data globally. Therefore, local learning is more direct which results in more accurate and efficient performance. For example, local learning used in learning classifiers from data, tries to employ a subset of input points around the separating hyperplane, while global learning tries to describe the overall phenomena utilizing all input points. Figure 2(a) illustrates the local learning. In this figure, the decision boundary is constructed only based on those filled points, while other points make no contributions to the classification plane (the decision planes are given based on the Gabriel Graph method [6, 7, 8], one of the local learning methods).

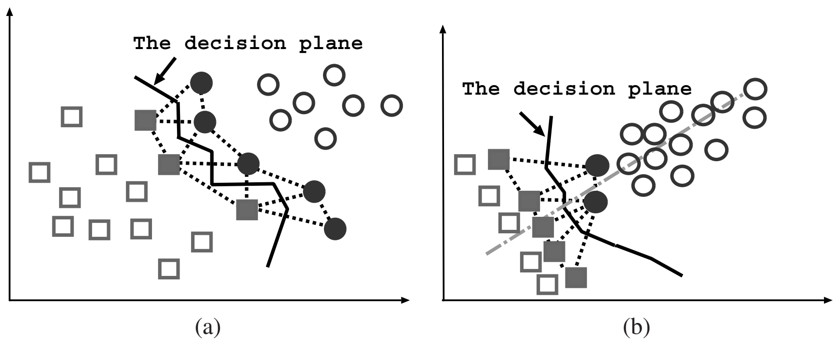


Fig. 2. (**a**) An illustration of local learning (also known as the Gabriel Graph classification). The decision boundary is just determined by some local points indicated as *filled points*. (**b**) An illustration on that local learning cannot grasp data trend

Although local learning appears to contain promising performance, it positions itself at the opposite extreme end to global learning. Employing only local information may lose the overall view of data. Local learning does not grasp the structure of the data, which may prove to be critical for guaranteeing better performance. This can be seen in the example as illustrated in Fig. 2(b). In this figure, the decision boundary (also constructed by the Gabriel Graph classification) is still determined by some local points indicated as filled points. Clearly, this boundary is myopic in nature and does not take into account the overall structure of the data. More specifically, the class associated with ◦'s is obviously more likely to scatter than the class associated with □'s in the axis indicated as dashed line. Therefore, instead of simply locating itself in the middle of the filled points, a more promising decision boundary should lie closer to the filled □'s than the filled ◦'s. A similar example can also be seen in Sect. 2 on a more principled local learning model, i.e., the current state-of-art-classifier, Support Vector Machines (SVM) [9]. Targeting at unifying this dichotomy, a hybrid learning is introduced in this chapter.

In summary, there are complementary advantages for both local learning and global learning. Global learning summarizes the data and provides the practitioners with knowledge on the structure of data, since with the precise modeling of phenomena, the observations can be accurately regenerated and therefore thoroughly studied or analyzed. However, this also presents

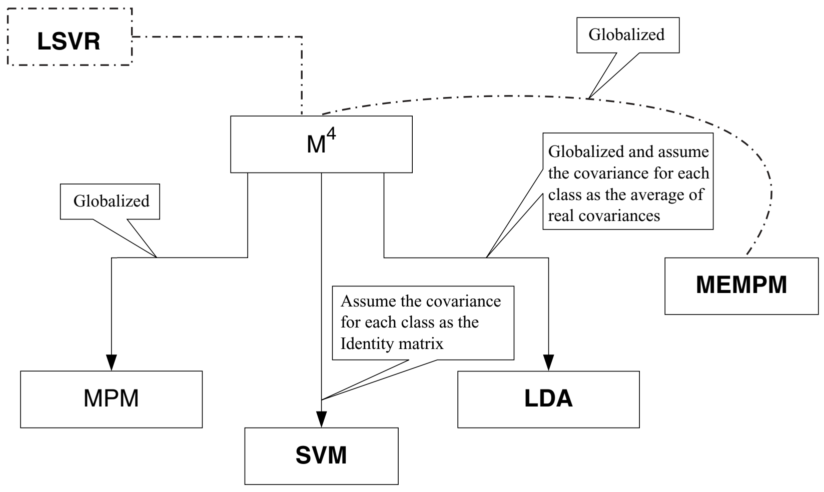

LSVR
Globalized
M^4
Globalized and assume the covariance for each class as the average of real covariances
Globalized
MEMPM
Assume the covariance for each class as the Identity matrix
MPM
LDA
SVM

Fig. 3. The relationship between M^4 and other related models

difficulties in how to choose a valid model to describe all the information. In comparison, local learning directly employs part of information critical for the specifically oriented tasks and does not assume a model for the data. Although demonstrated to be superior to global learning in various machine learning tasks, it misses some critical global information. The question here is thus, can reliable global information, independent of specific model assumptions, be combined into local learning from data? This question clearly motivates the development of hybrid learning for which Maxi-Min Margin Machine (M^4) is proposed.

As will be shown later in this chapter, M^4 has built various connections with both global learning and local learning models. As an overview, Fig. 3 briefly illustrates the relationship between M^4 and other models. When it is globalized, M^4 can change into a global learning model, the Minimum Probability Machine (MPM) model [10]. When some assumptions are made on the covariance matrices of data, it becomes another global learning model, the Linear Discriminant Analysis (LDA) [11]. The Support Vector Machine, one of the local learning models, is also its special case when certain conditions are satisfied. Moreover, when compared with a recently-proposed general global learning model, the Minimum Error Minimax Probability Machine (MEMPM) [12], M^4 can derive a very similar special case. Furthermore, a novel regression model, the Local Support Vector Regression (LSVR) [13] can also be connected with M^4.

The rest of this chapter is organized as follows. In the next section, we review the background of both global learning and local learning. In particular, we will provide some historical perspectives on three models, i.e., the Linear Discriminant Analysis, the Minimax Probability Machine, and the Support

Vector Machine. In Sect. 3, we introduce M^4 in details including its model definition, the geometrical interpretation, and its links with other models. Finally, we present remarks to conclude this chapter.

2 Background

In this section, we first review the background of global learning, then followed by the local learning models with emphasis on the current state-of-the-art classifier SVM. We then motivate the hybrid learning model, the Maxi-Min Margin Machine.

2.1 Global Learning

Traditional global learning methods with specific assumptions, i.e., generative learning, may not always coincide with data. Within the context of global learning, researchers begin to investigate approaches with no distributional assumptions. Following this trend, there are non-parametric methods, LDA, and a recently-proposed competitive model MPM. We will review them one by one.

In contrast with generative learning, non-parametric learning does not assume any specific global models before learning. Therefore, no risk will be taken on possible wrong assumptions on the data. Consequently, non-parametric learning appears to set a more valid foundation than generative learning models. One typical non-parametric learning model in the context of classification is the so-called Parzen Window estimation [14].

The Parzen Window estimation also attempts to estimate a density for the observed data. However it employs a different way from generative learning. Parzen window first defines an n-dimensional cell hypercube region R_N over each observation. By defining a window function,

$$w(\mathbf{u}) = \begin{cases} 1 & |\mathbf{u}_j| \le 1/2 \quad j = 1, 2, \ldots, n \\ 0 & \text{otherwise} , \end{cases} \tag{1}$$

the density is then estimated as

$$p_N(\mathbf{z}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h_N} w\left(\frac{\mathbf{z} - \mathbf{z}^i}{h_N}\right) , \tag{2}$$

where n is the data dimensionality, $\mathbf{z}_i$ for $1 \le i \le N$ represents N input data points, and h_N is defined as the length of the edge of R_N.

From the above, one can observe that Parzen Window puts a local density over each observation. The final density is then the statistical result of averaging all the local densities. In practice, the window function can actually be general functions including the most commonly-used Gaussian function.

These non-parametric methods make no underlying assumptions on data and appear to be more general in real cases. Using no parameters actually means using many "parameters" so that each parameter would not dominate other parameters (in the discussed models, the data points can be in fact considered as the "parameters"). In this way, if one parameter fails to work, it will not influence the whole system globally and statistically. However, using many "parameters" also results in serious problems. One of the main problems is that the density is overwhelmingly dependent on the training samples. Therefore, to generate an accurate density, the number of samples needs to be very large (much larger than would be required if we perform the estimation by generative learning approaches). Furthermore the number of data unfortunately increases exponentially with the dimension of data. Hence, it is usually hard to apply non-parametric learning in tasks with high-dimensional data. Another disadvantage caused is its severe requirement for the storage, since all the samples need to be saved beforehand in order to predict new data.

Instead of estimating an accurate distribution over data, an alternative approach is using some robust global information. The Linear Discriminant Analysis builds up a linear classifier by trying to minimize the intra-class distance while maximizing the inter-class distance. In this process, only up to the second order moments which are more robust with respect to the distribution, are adopted. Moreover, recent research has extended this linear classifier into nonlinear classification by using kernelization techniques [15]. In addition, a more recently-proposed model, the Minimax Probability Machine, goes further in this direction. Rather than constructing the decision boundary by estimating specific distributions, this approach exploits the worst-case distribution, which is distribution-free and more robust. With no assumptions on data, this model appears to be more valid in practice and is demonstrated to be competitive with the Support Vector Machine. Furthermore, Huang et al. [12] develop a superset of MPM, called Minimum Error Minimax Probability Machine, which achieves a worst-case distribution-free Bayes Optimal Classifier.

However, the problems for these models are that the robust estimation, e.g., the first and second order moments, may also be inaccurate. Considering specific data points, namely the local characteristic, seems to be necessary in this sense.

2.2 Local Learning

Local learning adopts a largely different way to construct classifiers. This type of learning is task-oriented. In the context of classification, only the final mapping function from the features $\mathbf{z}$ to the class variable c is crucial. Therefore, describing global information from data or explicitly summarizing a distribution, is an intermediate step. Hence the global learning scheme may be deemed wasteful or imprecise especially when the global information cannot be estimated accurately.

Alternatively, recent progress has suggested the local learning methodology. The family of approaches directly pin-points the most critical quantities for classification, while all other information less irrelevant to this purpose is simply omitted. Compared to global learning, this scheme assumes no model and also engages no explicit global information. Among this school of methods are neural networks [16, 17, 18, 19, 20, 21], Gabriel Graph methods [6, 7, 8], and large margin classifiers [5, 22, 23, 24] including Support Vector Machine, a state-of-the-art classifier which achieves superior performance in various pattern recognition tasks. In the following, we will focus on introducing SVM in details.

The Support Vector Machine is established based on minimizing the expected classification risk defined as follows:

$$\mathcal{R}(\mathbf{\Theta}) = \int_{\mathbf{z},\mathbf{c}} p(\mathbf{z}, c) l(\mathbf{z}, c, \mathbf{\Theta}) \ , \tag{3}$$

where $\mathbf{\Theta}$ represents the chosen model and the associate parameters, which are assumed to be the linear hyperplane in this chapter, and $l(\mathbf{z}, c, \mathbf{\Theta})$ is the loss function. Generally $p(\mathbf{z}, c)$ is unknown. Therefore, in practice, the above expected risk is often approximated by the so-called empirical risk:

$$\mathcal{R}_{\mathrm{emp}}(\mathbf{\Theta}) = \frac{1}{N} \sum_{j=1}^{N} l(\mathbf{z}^j, c^j, \mathbf{\Theta}) \ . \tag{4}$$

The above loss function describes the extent on how close the estimated class disagrees with the real class for the training data. Various metrics can be used for defining this loss function, including the 0–1 loss and the quadratic loss [25].

However, considering only the training data may lead to the over-fitting problem. In SVM, one big step in dealing with the over-fitting problem has been made, i.e., the margin between two classes should be pulled away in order to reduce the over-fitting risk. Figure 4 illustrates the idea of SVM. Two classes of data, depicted as circles and solid dots are presented in this figure. Intuitively observed, there are many decision hyperplanes, which can be adopted for separating these two classes of data. However, the one plotted in this figure is selected as the favorable separating plane, because it contains the maximum margin between the two classes. Therefore, in the objective function of SVM, a regularization term representing the margin shows up. Moreover, as seen in this figure, only those filled points, called *support vectors*, mainly determine the separating plane, while other points do not contribute to the margin at all. In other word, only several local points are critical for the classification purpose in the framework of SVM and thus should be extracted.

Actually, a more formal explanation and theoretical foundation can be obtained from the Structure Risk Minimization criterion [26, 27]. Therein,

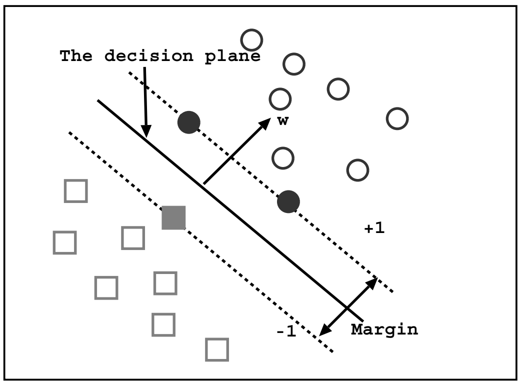


Fig. 4. An illustration of the Support Vector Machine

maximizing the margin between different classes of data is minimizing an upper bound of the expected risk, i.e., the VC dimension bound [27]. However, since the topic is out of the scope of this chapter, interested readers can refer to [9, 27].

2.3 Hybrid Learning

Local learning including SVM has demonstrated its advantages, such as its state-of-the-art performance (the lower generalization error), the optimal and unique solution, and the mathematical tractability [27]. However, it does discard many useful information from data, e.g., the structure information from data. An illustrative example has been seen in Fig. 2 in Sect. 1. In the current state-of-the-art classifier, i.e., SVM, similar problems also occur. This can be seen in Fig. 5. In this figure, the purpose is to separate two catergories of data $\mathbf{x}$ and $\mathbf{y}$, and the classification boundary is intuitively observed to be mainly determined by the dotted axis, i.e., the long axis of the $\mathbf{y}$ data (represented by $\square$'s) or the short axis of the $\mathbf{x}$ data (represented by $\circ$'s). Moreover, along this axis, the $\mathbf{y}$ data are more likely to scatter than the $\mathbf{x}$ data, since $\mathbf{y}$ contains a relatively larger variance in this direction. Noting this "global" fact, a good decision hyperplane seems reasonable to lie closer to the $\mathbf{x}$ side (see the dash-dot line). However, SVM ignores this kind of "global" information, i.e., the statistical trend of data occurrence: The derived SVM decision hyperplane (the solid line) lies unbiasedly right in the middle of two "local" points (the support vectors). The above considerations directly motivate the formulation of the Maxi-Min Margin Machine [28, 29].

3 Maxi-Min Margin Machine

In the following, we first present the scope and the notations. We then for the purpose of clarity, divide M^4 into *separable* and *nonseparable* categories,

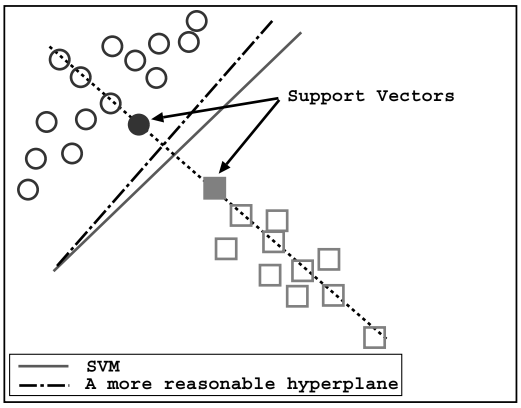


Fig. 5. A decision hyperplane with considerations of both local and global information

and introduce the corresponding hard-margin M^4 (linearly separable) and soft-margin M^4 (linearly non-separable) sequentially. Connections of the M^4 model with other models including SVM, MPM, LDA, and MEMPM will be provided in this section as well.

3.1 Scope and Notations

We only consider two-category classification tasks. Let a training data set contain two classes of samples, represented by $\mathbf{x}_i \in \mathbb{R}^n$ and $\mathbf{y}_j \in \mathbb{R}^n$ respectively, where $i = 1, 2, \ldots, N_{\mathbf{x}}$, $j = 1, 2, \ldots, N_{\mathbf{y}}$. The basic task here can be informally described as to find a suitable hyperplane $f(\mathbf{z}) = \mathbf{w}^T\mathbf{z} + b$ separating two classes of data as robustly as possible ($\mathbf{w} \in \mathbb{R}^n \backslash \{\mathbf{0}\}$, $b \in \mathbb{R}$, and $\mathbf{w}^T$ is the transpose of $\mathbf{w}$). Future data points $\mathbf{z}$ for which $f(\mathbf{z}) \geq 0$ are then classified as the class $\mathbf{x}$; otherwise, they are classified as the class $\mathbf{y}$. Throughout this chapter, unless we provide statements explicitly, bold typeface will indicate a vector or matrix, while normal typeface will refer to a scale variable or the component of the vectors.

3.2 Hard Margin Maxi-Min Margin Machine

Assuming the classification samples are separable, we first introduce the model definition and the geometrical interpretation. We then transform the model optimization problem into a sequential Second Order Cone Programming (SOCP) problem and discuss the optimization method.

The formulation for M^4 can be written as:

$$\max_{\rho,\mathbf{w}\neq\mathbf{0},b} \quad \rho \quad s.t. \tag{5}$$

$$\frac{(\mathbf{w}^T\mathbf{x}_i+b)}{\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}}} \geq \rho, \qquad i=1,2,\ldots,N_{\mathbf{x}}\ , \tag{6}$$

$$\frac{-(\mathbf{w}^T\mathbf{y}_j+b)}{\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}} \geq \rho, \quad j=1,2,\ldots,N_{\mathbf{y}}\ , \tag{7}$$

where Σ_x and Σ_y refer to the covariance matrices of the $\mathbf{x}$ and the $\mathbf{y}$ data, respectively.[1]

This model tries to *maximize* the margin defined as the *minimum* Mahalanobis distance for all training samples,[2] while simultaneously classifying all the data correctly. Compared to SVM, M^4 incorporates the data information in a global way; namely, the covariance information of data or the statistical trend of data occurrence is considered, while SVMs, including l_1-SVM [30] and l_2-SVM [5, 9],[3] simply discard this information or consider the same covariance for each class. Although the above decision plane is presented in a linear form, it has been demonstrated that the standard kernelization trick can be used to extend into the nonlinear decision boundary [12, 29]. Since the focus of this chapter lies at the introduction of M^4, we simply omit the elaboration of the kernelization.

A geometrical interpretation of M^4 can be seen in Fig. 6. In this figure, the $\mathbf{x}$ data are represented by the inner ellipsoid on the left side with its center as $\mathbf{x}_0$, while the $\mathbf{y}$ data are represented by the inner ellipsoid on the right side with its center as $\mathbf{y}_0$. It is observed that these two ellipsoids contain unequal covariances or risks of data occurrence. However, SVM does not consider this global information: Its decision hyperplane (the dotted line) locates unbiasedly in the middle of two support vectors (filled points). In comparison, M^4 defines the margin as a Maxi-Min Mahalanobis distance, which constructs a decision plane (the solid line) with considerations of both the local and global information: the M^4 hyperplane corresponds to the tangent line of two dashed ellipsoids centered at the support vectors (the local information) and shaped by the corresponding covariances (the global information).

3.2.1 Optimization Method

According to [12, 29], the optimization problem for the M^4 model can be cast as a sequential conic programming problem, or more specifically, a sequential SOCP problem. The strategy is based on the “Divide and Conquer” technique. One may note that in the optimization problem of M^4, if ρ is fixed to a

[1]For simplicity, we assume Σ_x and Σ_y are always positive definite. In practice, this can be satisfied by adding a small positive amount into their diagonal elements, which is widely used.

[2]This also motivates the name of our model.

[3]l_p-SVM means the “p-norm’ distance-based SVM.

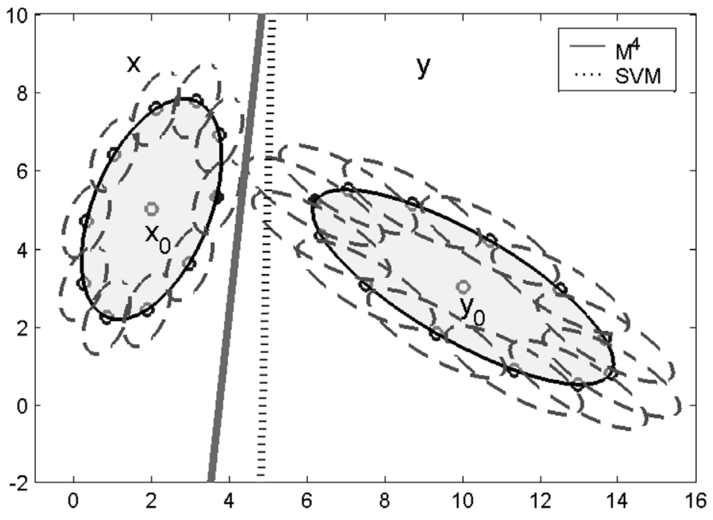


Fig. 6. A geometric interpretation of M^4. The M^4 hyperplane corresponds to the tangent line (*the solid line*) of two small dashed ellipsoids centered at the support vectors (the local information) and shaped by the corresponding covariances (the global information). It is thus more reasonable than the decision boundary calculated by SVM (*the dotted line*)

constant ρ_n, the problem is exactly changed to "conquer" the problem of checking whether the constraints of (6) and (7) can be satisfied. Moreover, as demonstrated by Theorem 1,[4] this "checking" procedure can be stated as an SOCP problem and can be solved in polynomial time. Thus the problem now becomes how ρ is set, which we can use "divide" to handle: If the constraints are satisfied, we can increase ρ_n accordingly; otherwise, we decrease ρ_n.

Theorem 1. *The problem of checking whether there exist* $\mathbf{w}$ *and* b *satisfying the following two sets of constraints (8) and (9) can be transformed as an SOCP problem, which can be solved in polynomial time,*

$$(\mathbf{w}^T\mathbf{x}_i + b) \geq \rho_n\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}}, \quad i = 1, \ldots, N_{\mathbf{x}}\ , \tag{8}$$

$$-(\mathbf{w}^T\mathbf{y}_j + b) \geq \rho_n\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}, \ j = 1, \ldots, N_{\mathbf{y}}\ . \tag{9}$$

Algorithm 3.2.1 lists the detailed step of the optimization procedure, which is also illustrated in Fig. 7.

In Algorithm 3.2.1, if a ρ satisfies the constraints of (6) and (7), we call it a feasible ρ; otherwise, we call it an infeasible ρ.

In practice, many SOCP programs, e.g., Sedumi [31], provide schemes to directly handle the above checking procedure.

[4]Detailed proof can be seen in [12, 29].

```
Get                    $\bar{\mathbf{x}}, \bar{\mathbf{y}}, \mathbf{\Sigma_x}, \mathbf{\Sigma_y}$ ;

Initialize             $\epsilon, \rho_0, \rho_{\max}$, where $\rho_0$ is a feasible $\rho$, and $\rho_{\max}$ is an
                       infeasible $\rho$ with $\rho_0 \le \rho_{\max}$ ;
Repeat                 $\rho_n = \frac{\rho_0+\rho_{\max}}{2}$ ;
                       Call the checking procedure to check whether $\rho_n$ is
                       feasible;
                       If $\rho_n$ is feasible
                           $\rho_0 = \rho_n$
                       Else
                           $\rho_{\max} = \rho_n$
until $|\rho_0 - \rho_{\max}| \le \epsilon$;

Assign                 $\rho = \rho_n$ ;
```

Algorithm 1: Optimization Algorithm of M^4

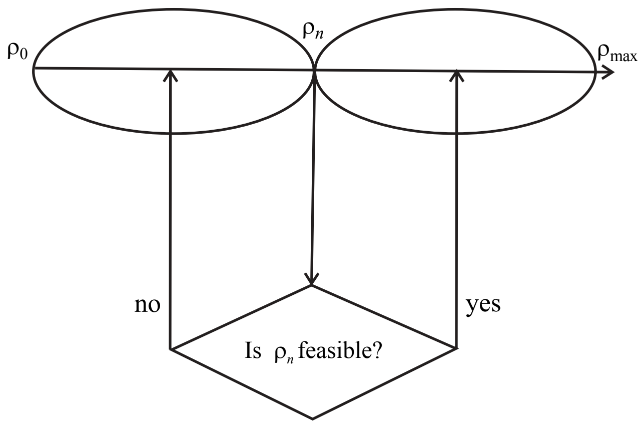


Fig. 7. A graph illustration on the optimization of M4

3.3 Time Complexity

We now analyze the time complexity of M^4. As indicated in [32], if the SOCP is solved based on interior-point methods, it contains a worst-case complexity of $O(n^3)$. If we denote the range of feasible ρ's as $L = \rho_{\max} - \rho_{\min}$ and the required precision as ε, then the number of iterations for M^4 is $\log(L/\varepsilon)$ in the worst case. Adding the cost of forming the system matrix (constraint matrix), which is $O(Nn^3)$ (N represents the number of training points), the total complexity would be $O(n^3 \log(L/\varepsilon) + Nn^3) \approx O(Nn^3)$, which is relatively large but can still be solved in polynomial time.[5]

[5]Note that the system matrix needs to be formed only once.

3.4 Soft Margin Maxi-Min Margin Machine

By introducing slack variables, the M^4 model can be extended to deal with nonseparable cases. This nonseparable version is written as follows:

$$\max_{\rho,\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\xi}} \quad \rho - C\sum_{k=1}^{N_{\mathbf{x}}+N_{\mathbf{y}}} \boldsymbol{\xi}_k \quad s.t. \tag{10}$$

$$(\mathbf{w}^T\mathbf{x}_i + b) \geq \rho\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}} - \boldsymbol{\xi}_i\ , \tag{11}$$

$$-(\mathbf{w}^T\mathbf{y}_j + b) \geq \rho\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}} - \boldsymbol{\xi}_{j+N_{\mathbf{x}}}\ , \tag{12}$$

$$\xi_k \geq 0\ ,$$

where $i = 1, \ldots, N_{\mathbf{x}}$, $j = 1, \ldots, N_{\mathbf{y}}$, and $k = 1, \ldots, N_{\mathbf{x}} + N_{\mathbf{y}}$. C is the positive penalty parameter and $\boldsymbol{\xi_k}$ is the slack variable, which can be considered as the extent how the training point $\mathbf{z}_k$ disobeys the ρ margin ($\mathbf{z}_k = \mathbf{x}_k$ when $1 \leq k \leq N_{\mathbf{x}}$; $\mathbf{z}_k = \mathbf{y}_{k-N_{\mathbf{y}}}$ when $N_{\mathbf{x}} + 1 \leq k \leq N_{\mathbf{x}} + N_{\mathbf{y}}$). Thus $\sum_{k=1}^{N_{\mathbf{x}}+N_{\mathbf{y}}} \boldsymbol{\xi}_k$ can be conceptually regarded as the training error or the empirical error. In other words, the above optimization achieves maximizing the minimum margin while minimizing the total training error. The above optimization can further be solved based on a linear search problem [33] combined with the second order cone programming problem [12, 29].

3.5 A Unified Framework

In this section, connections between M^4 and other models are established. More specifically, SVM, MPM, and LDA are actually special cases of M^4 when certain assumptions are made on the data. M^4 therefore represents a unified framework of both global models, e.g., LDA and MPM, and a local model, i.e., SVM.

Corollary 1 M^4 *reduces to the Minimax Probability Machine, when it is "globalized".*

This can be easily seen by expanding and adding the constraints of (6) together. One can immediately obtain the following:

$$\mathbf{w}^T\sum_{i=1}^{N_{\mathbf{x}}}\mathbf{x}_i + N_{\mathbf{x}}b \geq N_{\mathbf{x}}\rho\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}}\ ,$$

$$\Rightarrow \mathbf{w}^T\overline{\mathbf{x}} + b \geq \rho\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}}\ , \tag{13}$$

where $\overline{\mathbf{x}}$ denotes the mean of the $\mathbf{x}$ training data.

Similarly, from (7) one can obtain:

$$
\begin{aligned}
&-\left(\mathbf{w}^T \sum_{j=1}^{N_{\mathbf{y}}} \mathbf{y}_j + N_{\mathbf{y}} b\right) \geq N_{\mathbf{y}} \rho \sqrt{\mathbf{w}^T \Sigma_{\mathbf{y}} \mathbf{w}}\ , \\
&\Rightarrow -(\mathbf{w}^T \overline{\mathbf{y}} + b) \geq \rho \sqrt{\mathbf{w}^T \Sigma_{\mathbf{y}} \mathbf{w}}\ , \qquad (14)
\end{aligned}
$$

where $\overline{\mathbf{y}}$ denotes the mean of the $\mathbf{y}$ training data.

Adding (13) and (14), one can obtain:

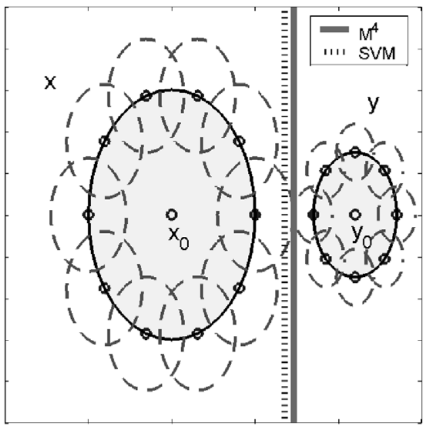
$$
\begin{aligned}
&\max_{\rho, \mathbf{w} \neq \mathbf{0}} \quad \rho \quad \text{s.t.} \\
&\mathbf{w}^T (\overline{\mathbf{x}} - \overline{\mathbf{y}}) \geq \rho \left(\sqrt{\mathbf{w}^T \Sigma_{\mathbf{x}} \mathbf{w}} + \sqrt{\mathbf{w}^T \Sigma_{\mathbf{y}} \mathbf{w}} \right)\ . \qquad (15)
\end{aligned}
$$

The above optimization is exactly the MPM optimization [10]. Note, however, that the above procedure is irreversible. This means the MPM is a special case of M^4. In MPM, since the decision is completely determined by the global information, i.e., the mean and covariance matrices [10], the estimates of mean and covariance matrices need to be reliable to assure an accurate performance. However, it cannot always be the case in real-world tasks. On the other hand, M^4 solves this problem in a natural way, because the impact caused by inaccurately estimated mean and covariance matrices can be neutralized by utilizing the local information, namely by satisfying those constraints of (6) and (7) for each local data point.

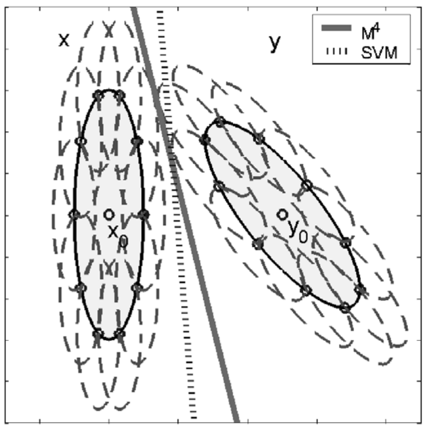
Corollary 2 M^4 *reduces to the Minimax Probability Machine, when* $\Sigma_{\mathbf{x}} = \Sigma_{\mathbf{y}} = \Sigma = \mathbf{I}$.

Intuitively, as two covariance matrices are assumed to be equal, the Mahalanobis distance changes to the Euclidean distance as used in standard SVM. The M^4 model will naturally reduce to the SVM model (refer to [12, 29] for a detailed proof). From the above, we can consider that two assumptions are implicitly made by SVM: One is the assumption on data "orientation" or data shape, i.e., $\Sigma_{\mathbf{x}} = \Sigma_{\mathbf{y}} = \Sigma$, and the other is the assumption on data "scattering magnitude" or data compactness, i.e., $\Sigma = \mathbf{I}$. However, these two assumptions are inappropriate. We demonstrate this in Fig. 8(a) and Fig. 8(b). We assume the orientation and the magnitude of each ellipsoid represent the data shape and compactness, respectively, in these figures.

Figure 8(a) plots two types of data with the same data orientations but different data scattering magnitudes. It is obvious that, by ignoring data scattering, SVM is improper to locate itself unbiasedly in the middle of the support vectors (filled points), since $\mathbf{x}$ is more possible to scatter in the horizontal axis. Instead, M^4 is more reasonable (see the solid line in this figure). Furthermore, Fig. 8(b) plots the case with the same data scattering magnitudes but different data orientations. Similarly, SVM does not capture the orientation information. In comparison, M^4 grasps this information and demonstrates a more suitable decision plane: M^4 represents the tangent line between two small



(a)



(b)

Fig. 8. An illustration on the connections between SVM, and M^4. (**a**) demonstrates SVM omits the data compactness information. (**b**) demonstrates SVM discards the data orientation information

dashed ellipsoids centered at the support vectors (filled points). Note that SVM and M^4 do not need to generate the same support vectors. In Fig. 8(b), M^4 contains the above two filled points as support vectors, whereas SVM has all the three filled points as support vectors.

Corollary 3 M^4 *reduces to the LDA model, when it is "globalized" and assumes* $\Sigma_{\mathbf{x}} = \Sigma_{\mathbf{y}} = (\widehat{\Sigma_{\mathbf{x}}} + \widehat{\Sigma_{\mathbf{y}}})/2$*, where* $\widehat{\Sigma_{\mathbf{x}}}$ *and* $\widehat{\Sigma_{\mathbf{y}}}$ *are estimates of covariance matrices for the class* $\mathbf{x}$ *and* $\mathbf{y}$ *respectively.*

If we change the denominators in (6) and (7) as $\sqrt{\mathbf{w}^T\widehat{\Sigma_{\mathbf{x}}}\mathbf{w} + \mathbf{w}^T\widehat{\Sigma_{\mathbf{y}}}\mathbf{w}}$, the optimization can be changed as:

$$\max_{\rho,\mathbf{w}\neq\mathbf{0},b} \quad \rho \quad s.t. \tag{16}$$

$$\frac{(\mathbf{w}^T\mathbf{x}_i + b)}{\sqrt{\mathbf{w}^T\widehat{\Sigma_{\mathbf{x}}}\mathbf{w} + \mathbf{w}^T\widehat{\Sigma_{\mathbf{y}}}\mathbf{w}}} \geq \rho \ , \tag{17}$$

$$\frac{-(\mathbf{w}^T\mathbf{y}_j + b)}{\sqrt{\mathbf{w}^T\widehat{\Sigma_{\mathbf{x}}}\mathbf{w} + \mathbf{w}^T\widehat{\Sigma_{\mathbf{y}}}\mathbf{w}}} \geq \rho \ , \tag{18}$$

where $i = 1, \ldots, N_{\mathbf{x}}$ and $j = 1, \ldots, N_{\mathbf{y}}$. If the above two sets of constraints for $\mathbf{x}$ and $\mathbf{y}$ are "globalized" via a procedure similar to that in MPM, the above optimization problem is easily verified to be the following optimization:

$$\max_{\rho,\mathbf{w}\neq\mathbf{0},b} \quad \rho \quad s.t.$$

$$\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}}) \geq \rho\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w} + \mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}} \ . \tag{19}$$

Note that (19) can be changed as $\rho \leq \frac{|\mathbf{w}^T(\overline{\mathbf{x}}-\overline{\mathbf{y}})|}{\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}+\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}}$, which is exactly the optimization of the LDA.

Corollary 4 *When a "globalized" procedure is performed on the soft margin version,* M^4 *reduces to a large margin classifier as follows:*

$$\max_{\mathbf{w}\neq\mathbf{0},b} \quad \theta t + (1-\theta)s \quad s.t. \tag{20}$$

$$\frac{\mathbf{w}^T\overline{\mathbf{x}}+b}{\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}}} \geq t \ , \tag{21}$$

$$-\frac{\mathbf{w}^T\overline{\mathbf{y}}+b}{\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}} \geq s \ . \tag{22}$$

We can see that the above formula optimize a very similar form as the MEMPM model except that (20) changes to $\min_{\mathbf{w}\neq\mathbf{0},b}\theta\frac{t^2}{1+t^2}+(1-\theta)\frac{s^2}{1+s^2}$ [12]. In MEMPM, $\frac{t^2}{1+t^2}$ ($\frac{s^2}{1+s^2}$) (denoted as α (β)) represents the worst-case accuracy for the classification of future $\mathbf{x}$ ($\mathbf{y}$) data. Thus MEMPM maximizes the weighted accuracy on the future data. In M^4, s and t represent the corresponding margin, which is defined as the distance from the hyperplane to the class center. Therefore, it represents the weighted maximum margin machine in this sense. Moreover, since the conversion function of $g(u) = \frac{u^2}{1+u^2}$ increases monotonically with u, maximizing the above formulae contains a physical meaning similar to the optimization of MEMPM. For the proof, please refer to [12, 29].

3.5.1 Relationship with Local Support Vector Regression

A recently proposed promising model, the Local Support Vector Regression [13], can also be linked with M^4. In regression, the objective is to learn a model from a given data set, $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, and then based on the learned model to make accurate predictions of y for future values of $\mathbf{x}$. The LSVR optimization is formulated as follows:

$$\min_{\mathbf{w},b,\xi_i,\xi_i^*} \frac{1}{N}\sum_{i=1}^{N}\sqrt{\mathbf{w}^T\Sigma_i\mathbf{w}} + C\sum_{i=1}^{N}(\xi_i+\xi_i^*) \ , \tag{23}$$

$$\begin{aligned} \text{s.t.} \ \ & y_i - (\mathbf{w}^T\mathbf{x}_i + b) \leq \epsilon\sqrt{\mathbf{w}^T\Sigma_i\mathbf{w}} + \xi_i \ , \\ & (\mathbf{w}^T\mathbf{x}_i + b) - y_i \leq \epsilon\sqrt{\mathbf{w}^T\Sigma_i\mathbf{w}} + \xi_i^* \ , \\ & \xi_i \geq 0, \quad \xi_i^* \geq 0, \quad i = 1, \ldots, N \ , \end{aligned} \tag{24}$$

where ξ_i and ξ_i^* are the corresponding up-side and the down-side errors at the i-th point, respectively. ϵ is a positive constant, which defines the margin width. Σ_i is the covariance matrix formed by the i-th data point and those data points close to it. In the state-of-the-art regression model, namely, the support vector regression [27, 34, 35, 36], the margin width ϵ is fixed. As a comparison in LSVR, this width is adapted automatically and locally with respect to the data volatility. More specifically, suppose $y_i = \mathbf{w}^T\mathbf{x}_i + b$ and $\bar{y_i} = \mathbf{w}^T\bar{\mathbf{x}}_\mathbf{i} + b$. The variance around the i-th data point is written as $\Delta_i = \frac{1}{2k+1}\sum_{j=-k}^{k}(y_{i+j} - \bar{y_i})^2 = \frac{1}{2k+1}\sum_{j=-k}^{k}(\mathbf{w}^T(\mathbf{x}_{i+j} - \bar{\mathbf{x}}_i))^2 = \mathbf{w}^T\Sigma_i\mathbf{w}$, where $2k$ is the number of data points closest to the i-th data point. Therefore, $\Delta_i = \mathbf{w}^T\Sigma_i\mathbf{w}$ actually captures the volatility in the local region around the i-th data point. LSVR can systematically and automatically vary the tube: If the i-th data point lies in the area with a larger variance of noise, it will contribute to a larger $\epsilon\sqrt{\mathbf{w}^T\Sigma_i\mathbf{w}}$ or a larger local margin. This will result in reducing the impact of the noise around the point; on the other hand, in the case that the i-th data point is in the region with a smaller variance of noise, the local margin (tube), $\epsilon\sqrt{\mathbf{w}^T\Sigma_i\mathbf{w}}$, will be smaller. Therefore, the corresponding point would contribute more in the fitting process.

The LSVR model can be considered as an extension of M^4 into the regression task. Within the framework of classification, M^4 considers different data trends for different classes. Analogously, in the novel LSVR model, we allow different data trends for different regions, which is more suitable for the regression purpose.

4 Conclusion

We present a unifying theory of the Maxi-Min Margin Machine (M^4) that combines two schools of learning thoughts, i.e., local learning and global learning. This hybrid model is shown to subsume both global learning models, i.e., the Linear Discriminant Analysis and the Minimax Probability Machine, and a local learning model, the Support Vector Machine. Moreover, it can be linked with a worst-case distribution-free Bayes optimal classifier, the Minimum Error Minimax Probability Machine and a promising regression model, the Local Support Vector Regression. Historical perspectives, the geometrical interpretation, the detailed optimization algorithm, and various theoretical connections are provided to introduce this novel and promising framework.

Acknowledgements

The work described in this paper was fully supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK4182/03E and Project No. CUHK4235/04E).

References

1. Grzegorzewski, P., Hryniewicz, O., and Gil, M. (2002). Soft methods in probability, statistics and data analysis, Physica-Verlag, Heidelberg; New York. 114

2. Duda, R. and Hart, P. (1973). Pattern classification and scene analysis: John Wiley & Sons. 114
3. Girosi, F. (1998). An equivalence between sparse approximation and support vector machines, *Neural Computation* **10(6)**, 1455–1480. 114
4. Schölkopf, B. and Smola, A. (2002). Learning with Kernels, MIT Press, Cambridge, MA. 114
5. Smola, A. J., Bartlett, P. L., Scholkopf, B., and Schuurmans, D. (2000). Advances in Large Margin Classifiers, The MIT Press. 114, 119, 122
6. Barber, C. B., Dobkin, D. P., and Huhanpaa, H. (1996). The Quickhull Algorithm for Convex Hulls, *ACM Transactions on Mathematical Software* **22(4)**, 469–483. 115, 119
7. J. W. Jaromczyk, G. T. (1992). Relative Neighborhood Graphs And Their Relatives, *Proceedings IEEE* **80(9)**, 1502–1517. 115, 119
8. Zhang, W. and King, I. (2002). A study of the relationship between support vector machine and Gabriel graph, in Proceedings of IEEE World Congress on Computational Intelligence – International Joint Conference on Neural Networks. 115, 119
9. Vapnik, V. N. (1998). Statistical Learning Theory, John Wiley & Sons. 115, 120, 122
10. Lanckriet, G. R. G., Ghaoui, L. E., Bhattacharyya, C., and Jordan, M. I. (2002). A Robust Minimax Approach to Classification, *Journal of Machine Learning Research* **3**, 555–582. 116, 126
11. Fukunaga, K. (1990). Introduction to Statistical Pattern Recognition, Academic Press, San Diego 2nd edition. 116
12. Huang, K., Yang, H., King, I., Lyu, M. R., and Chan, L. (2004). The Minimum Error Minimax Probability Machine, Journal of Machine Learning Research, 5:1253–1286, October 2004. 116, 118, 122, 123, 125, 126, 128
13. Huang, K., Yang, H., King, I., and Lyu, M. R. (2004). Varying the Tube: A Local Support Vector Regression Model, *Technique Report, Dept. of Computer Science and Engineering, The Chinese Univ. of Hong Kong*. 116, 128
14. Duda, R. O., Hart, P. E., and Stork, D. G. (2000). Pattern Classification, John Wiley & Sons. 117
15. Mika, S., Ratsch, G., Weston, J., Scholkopf, B., and Muller, K.-R. (1999). Fisher discriminant analysis with kernels, *Neural Networks for Signal Processing IX* pp. 41–48. 118
16. Anand, R., Mehrotram, G. K., Mohan, K. C., and Ranka, S. (1993). An improved alogrithm for Neural Network Classification of Imbalance Training Sets, *IEEE Transactions on Neural Networks* **4(6)**, 962–969. 119
17. Fausett, L. (1994). Fundamentals of Neural Networks., New York: Prentice Hall. 119
18. Haykin, S. (1994). Neural Networks: A Comprehensive Foundation., New York: Macmillan Publishing. 119
19. Mehra, P. and Wah., B. W. (1992). Artificial neural networks: concepts and theory, Los Alamitos, California: IEEE Computer Society Press. 119
20. Patterson, D. (1996). Artificial Neural Networks., Singapore: Prentice Hall. 119
21. Ripley, B. (1996). Pattern Recognition and Neural Networks, Press Syndicate of the University of Cambridge. 119
22. Cristianini, N. and Shawe-Taylor, J. (2000). An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, Cambridge, U.K.; New York. 119
23. Schölkopf, B. Burges, C. and Smola, A. (ed.) (1999). Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, Massachusetts. 119

24. Schölkopf, B. and Smola, A. (ed.) (2002). Learning with kernels: support vector machines, regularization, optimization and beyond, MIT Press, Cambridge, Massachusetts. 119
25. Trivedi, P. K. (1978). Estimation of a Distributed Lag Model under Quadratic Loss, *Econometrica* **46(5)**, 1181–1192. 119
26. Burges, C. J. C. (1998). A Tutorial on Support Vector Machines for Pattern Recognition, *Data Mining and Knowledge Discovery* **2(2)**, 121–167. 119
27. Vapnik, V. N. (1999). The Nature of Statistical Learning Theory, Springer, New York 2nd edition. 119, 120, 129
28. Huang, K., Yang, H., King, I., and Lyu, M. R. (2004). Learning large margin classifiers locally and globally, in the Twenty-First International Conference on Machine Learning (ICML-2004): pp. 401–408. 120
29. Huang, K., Yang, H., King, I., and Lyu, M. R. (2004). Maxi-Min Margin Machine: Learning large margin classifiers globally and locally, Journal of Machine Learning, submitted. 120, 122, 123, 125, 126, 128
30. Zhu, J., Rosset, S., Hastie, T., and Tibshirani, R. (2003). 1-norm Support Vector Machines, In Advances in Neural Information Processing Systems (NIPS 16). 122
31. Sturm, J. F. (1999). Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones, *Optimization Methods and Software* **11**, 625–653. 123
32. Lobo, M., Vandenberghe, L., Boyd, S., and Lebret, H. (1998). Applications of second order cone programming, *Linear Algebra and its Applications* **284**, 193–228. 124
33. Bertsekas, D. P. (1999). Nonlinear Programming, Athena Scientific, Belmont, Massashusetts 2nd edition. 125
34. Drucker, H., Burges, C., Kaufman, L., Smola, A., and Vapnik, V. N. (1997). Support Vector Regression Machines, in Michael C. Mozer, Michael I. Jordan, and Thomas Petsche (ed.), Advances in Neural Information Processing Systems, volume **9**, The MIT Press pp. 155–161. 129
35. Gunn, S. (1998). Support vector machines for classification and regression Technical Report NC2-TR-1998-030 Faculty of Engineering and Applied Science, Department of Electronics and Computer Science, University of Southampton. 129
36. Smola, A. and Schölkopf, B. (1998). A tutorial on support vector regression Technical Report NC2-TR-1998-030 NeuroCOLT2. 129