Learning Large Margin Classifiers Locally and Globally

Kaizhu Huang KZHUANG@CSE.CUHK.EDU.HK
Haiqin Yang HQYANG@CSE.CUHK.EDU.HK
Irwin King KING@CSE.CUHK.EDU.HK
Michael R. Lyu LYU@CSE.CUHK.EDU.HK
Department of Computer Science and Engineering,
The Chinese University of Hong Kong,
Shatin, N.T., Hong Kong

Abstract

A new large margin classifier, named Maxi-Min Margin Machine (M^4) is proposed in this paper. This new classifier is constructed based on both a "local" and a "global" view of data, while the most popular large margin classifier, Support Vector Machine (SVM) and the recently-proposed important model, Minimax Probability Machine (MPM) consider data only either locally or globally. This new model is theoretically important in the sense that SVM and MPM can both be considered as its special case. Furthermore, the optimization of M^4 can be cast as a sequential conic programming problem, which can be solved efficiently. We describe the M^4 model definition, provide a clear geometrical interpretation, present theoretical justifications, propose efficient solving methods, and perform a series of evaluations on both synthetic data sets and real world benchmark data sets. Its comparison with SVM and MPM also demonstrates the advantages of our new model.

1. Introduction

Recently, learning large margin classifiers (Smola et al., 2000) has become an active research topic. Support Vector Machine (SVM) (Vapnik, 2000), the most famous one of them, achieves a great success in machine learning and pattern recognition. SVM aims to find a hyperplane, which can separate two classes of data with the maximal margin. However, this margin is defined in a "local" way, i.e., the margin is exclusively determined by some critical points, which are called support vectors, whereas all other points are totally irrelevant to the decision hyperplane. Although this scheme is both theoretically and empirically demonstrated to be powerful, it actually discards the global information of data. An illustration example can be seen in Figure 1. In this figure, the classification boundary is intuitively observed to be mainly determined by the dotted axis, i.e., the long axis of the **y** data (represented by □'s) or the short axis of the **x** data (represented by o's). Moreover, along this axis, the **y** data are more possible to scatter than the **x** data, since **y** contains a relatively larger variance in this direction. Noting this "global" fact, a good decision hyperplane seems reasonable to lie closer to the **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).

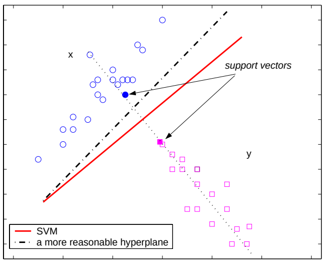


Figure 1. A decision hyperplane with considerations of both local and global information.

Motivated from this important observation, we propose Maxi-Min Margin Machine (M^4) to simultaneously consider data in both a local and a global fashion. Interestingly, as we show later, M^4 is actually a unified model of SVM and another recently-

Appearing in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004.

proposed promising model Minimax Probability Machine (MPM) (Lanckriet et al., 2002). Moreover, based on our proposed local and global view of data, another popular model, Linear Discriminant Analysis (LDA) (Fukunaga, 1990) can easily be interpreted and extended as well. Another critical feature of M^4 is that, it can be cast as a sequential conic programming problem (Pruessner, 2003), or more specifically, a sequential Second Order Cone Programming (SOCP) problem (Lobo et al., 1998), which can be solved efficiently.

This paper is organized as follows. In the next section, we introduce the M^4 model in detail, including its model definition, the geometrical interpretation, connections with other models, and the associated solving methods. Following that, we evaluate this novel model in Section 4. Finally, we conclude this paper in Section 5.

2. Maxi-Min Margin Machine

In the following, we first introduce the notations used in this paper. We then, for the purpose of clarity, divide M^4 into separable and nonseparable categories, and introduce the corresponding models sequently.

2.1. Notations

We only consider two-category classification tasks. Assuming a training data set contains 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 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 $\mathbf{y}$.

2.2. Separable Case

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 SOCP problem and discuss the detailed optimization method.

2.2.1. Model Definition

The formulation for M^4 can be written as:

$$\max_{\rho, \mathbf{w} \neq \mathbf{0}, b} \quad \rho \quad s.t. \tag{1}$$

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

$$\frac{-(\mathbf{w}^T\mathbf{y}_j + b)}{\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}} \geq \rho, \quad j = 1, 2, \ldots, N_{\mathbf{y}}\ , \tag{3}$$

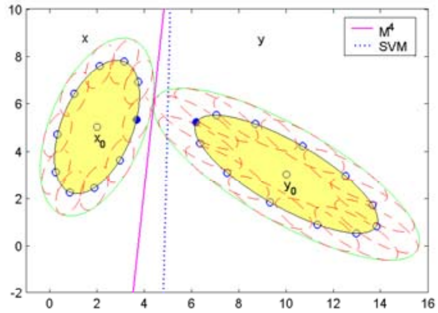


Figure 2. A geometric interpretation of M^4.

where $\Sigma_{\mathbf{x}}$ and $\Sigma_{\mathbf{y}}$ refer to the covariance matrices of the $\mathbf{x}$ and the $\mathbf{y}$ data, respectively.

This model tries to *maximize* the margin defined as the *minimum* Mahalanobis distance for all training samples,[1] 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, l_2-SVM, and l_∞-SVM (l_p-SVM means the "p-norm" distance-based SVM) (Smola et al., 2000), simply discard this information or consider the same covariance for each class.

A geometrical interpretation of M^4 can be seen in Figure 2. 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 blue 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 thus constructs a decision plane (the solid magenta 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).

2.2.2. Optimization Method

We will in the following show how the above problem can be cast as a sequential conic programming problem, or more specifically, a sequential SOCP problem.

Our strategy is based on the "Divide and Conquer"

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

technique. One may note that in the optimization problem of M^4, if ρ is fixed to a constant ρ_n, the problem is exactly changed to "conquer" the problem of checking whether the constraints of (2) and (3) can be satisfied. Moreover, as will be demonstrated shortly, this "checking" procedure can be stated as an SOCP problem. 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.

We detail this solving technique in the following two steps:

1. **Divide:** Set $\rho_n = (\rho_0 + \rho_m)/2$, where ρ_0 is a feasible ρ, ρ_m is an infeasible ρ, and $\rho_0 \le \rho_m$.

2. **Conquer:** Call the Modified Second Order Cone Programming (MSOCP) procedure elaborated in the following to check whether ρ_n is a feasible ρ. If yes, set $\rho_0 = \rho_n$; otherwise, set $\rho_m = \rho_n$;

In the above, if a ρ satisfies the constraints of (2) and (3), we call it a feasible ρ; otherwise, we call it an infeasible ρ. These two steps are iterated until $|\rho_0 - \rho_m|$ is less than a small positive value.

The MSOCP procedure is introduced in the following. We reformulate the constraints of (2) and (3) as follows:

$$
\begin{aligned}
(\mathbf{w}^T\mathbf{x}_i + b) &\ge \rho_n\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}},\ i = 1,\dots,N_{\mathbf{x}}\ ,\\
-(\mathbf{w}^T\mathbf{y}_j + b) &\ge \rho_n\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}},\ j = 1,\dots,N_{\mathbf{y}}\ .
\end{aligned}
$$

Our task here is to check whether there exist a $\mathbf{w}$ and a b satisfying the above two constraints, which are obviously the forms of the second order cones (here ρ_n is a constant). Actually, many SOCP programs, e.g., Sedumi (Sturm, 1999), provide schemes to directly handle the above checking procedure. However, to make it clear, we elaborate in the following how this checking problem can be transformed as an SOCP optimization problem.

Introducing dummy variables $\boldsymbol{\tau}$, we rewrite the above checking problem into an equivalent optimization problem:

$$
\begin{aligned}
&\max_{\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\tau}} \quad \{\min_{k=1}^{N_{\mathbf{x}}+N_{\mathbf{y}}} \quad \boldsymbol{\tau}_k\} \quad s.t.\\
&(\mathbf{w}^T\mathbf{x}_i + b) \ge \rho_n\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}} - \boldsymbol{\tau}_i,\\
&-(\mathbf{w}^T\mathbf{y}_j + b) \ge \rho_n\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}} - \boldsymbol{\tau}_{j+N_{\mathbf{x}}}\ ,
\end{aligned}
$$

where $i = 1,\dots,N_{\mathbf{x}}$ and $j = 1,\dots,N_{\mathbf{y}}$. By checking whether the minimum $\boldsymbol{\tau}_k$ at the optimum point is positive, we can know whether the constraints of (2) and (3) can be satisfied.

We can further introduce another dummy variable and transform the above problem into an SOCP problem:

$$
\begin{aligned}
&\max_{\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\tau},\eta} \quad \eta \quad s.t.\\
&(\mathbf{w}^T\mathbf{x}_i + b) \ge \rho_n\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}} - \boldsymbol{\tau}_i\ ,\\
&-(\mathbf{w}^T\mathbf{y}_j + b) \ge \rho_n\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}} - \boldsymbol{\tau}_{j+N_{\mathbf{x}}}\ ,\\
&\eta \le \boldsymbol{\tau}_k\ ,
\end{aligned}
$$

where $i = 1,\dots,N_{\mathbf{x}}$, $j = 1,\dots,N_{\mathbf{y}}$, and $k = 1,\dots,N_{\mathbf{x}} + N_{\mathbf{y}}$. By checking whether the optimal η is greater than 0, we can immediately know whether there exist a $\mathbf{w}$ and a b satisfying the constraints of (2) and (3). Moreover, the above optimization is easily verified to be the standard SOCP form, since the optimization function is a linear form and the constraints are either linear or the typical second order conic constraints.

We now analyze the time complexity of M^4. As indicated in (Lobo et al., 1998), 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 $O(\log(L/\varepsilon)n^3 + Nn^3) \approx O(Nn^3)$, which is relatively large but can still be solved in polynomial time.[2]

2.3. Connections with Other Models

In this section, we establish connections between M^4 and other models. We show that SVM and MPM are actually special cases of our model. Moreover, LDA can be interpreted and extended according to our local and global views of data.

2.3.1. Connection with Minimax Probability Machine

If one expands the constraints of (2) and add all of them together, one can immediately obtain the following:

$$
\mathbf{w}^T\overline{\mathbf{x}} + b \ge \rho\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}}\ , \tag{4}
$$

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

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

Similarly, from (3) one can obtain:

$$-(\mathbf{w}^T\overline{\mathbf{y}}+b) \geq \rho\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}\ , \tag{5}$$

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

Adding (4) and (5), one can obtain:

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

$$\mathbf{w}^T(\overline{\mathbf{x}}-\overline{\mathbf{y}}) \geq \rho(\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}}+\sqrt{\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}})\ . \tag{6}$$

The above optimization is exactly the MPM optimization (Lanckriet et al., 2002). Note, however, that the above procedure cannot be reversed. This means the MPM is a special case of M^4.

Remarks: In MPM, since the decision is completely determined by the global information, namely, the mean and covariance matrices (Lanckriet et al., 2002),[3] to assure an accurate performance, the estimates of mean and covariance matrices need to be reliable. However, it cannot always be the case in real world tasks. On the other hand, M^4 seems to solve 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 (2) and (3) for each local data point. This is also demonstrated in the later experiment.

2.3.2. Connection with Support Vector Machine

If one assumes $\Sigma_{\mathbf{x}}=\Sigma_{\mathbf{y}}=\Sigma$, the optimization of M^4 can be changed as:

$$\begin{aligned}\max_{\rho,\mathbf{w}\neq\mathbf{0},b} \quad & \rho \quad s.t.\\ & (\mathbf{w}^T\mathbf{x}_i+b) \geq \rho\sqrt{\mathbf{w}^T\Sigma\mathbf{w}}\ ,\\ & -(\mathbf{w}^T\mathbf{y}_j+b) \geq \rho\sqrt{\mathbf{w}^T\Sigma\mathbf{w}}\ ,\end{aligned}$$

where $i=1,\ldots,N_{\mathbf{x}}$ and $j=1,\ldots,N_{\mathbf{y}}$.

Observing that the magnitude of $\mathbf{w}$ will not influence the optimization, without loss of generality, one can further assume $\rho\sqrt{\mathbf{w}^T\Sigma\mathbf{w}}=1$. Therefore the optimization can be changed as:

$$\min_{\mathbf{w}\neq\mathbf{0},b} \quad \mathbf{w}^T\Sigma\mathbf{w} \quad s.t. \tag{7}$$

$$(\mathbf{w}^T\mathbf{x}_i+b) \geq 1\ , \tag{8}$$

$$-(\mathbf{w}^T\mathbf{y}_j+b) \geq 1\ , \tag{9}$$

where $i=1,\ldots,N_{\mathbf{x}}$ and $j=1,\ldots,N_{\mathbf{y}}$.

A special case of the above with $\Sigma=\mathbf{I}$ is precisely the optimization of SVM, where $\mathbf{I}$ is the identity matrix.

Remarks: In the above, 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 Figure 3(a) and Figure 3(b). We assume the orientation and the magnitude of each ellipsoid represent the data shape and compactness, respectively, in these figures.

Figure 3(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 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, Figure 3(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 dashed ellipsoids centered at the support vectors (filled points). Note that SVM and M^4 do not need to achieve the same support vectors. In Figure 3(b), M^4 contains the above two filled points as support vectors, whereas SVM has all the three filled points as support vectors.

2.3.3. Link with Linear Discriminant Analysis

LDA, an important and popular method, is used widely in constructing decision hyperplanes and reducing the feature dimensionality. In the following discussion, we mainly consider its application as a classifier. LDA involves solving the following optimization problem: $\max_{\mathbf{w}\neq\mathbf{0}}\frac{|\mathbf{w}^T(\overline{\mathbf{x}}-\overline{\mathbf{y}})|}{\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}+\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}}$. Similar to MPM, LDA also focuses on using the global information rather than considering data both locally and globally. We now show that LDA can be modified to consider data both locally and globally.

If one changes the denominators in (2) and (3) as $\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}+\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}$, the optimization can be changed as:

$$\max_{\rho,\mathbf{w}\neq\mathbf{0},b} \quad \rho \quad s.t. \tag{10}$$

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

$$\frac{-(\mathbf{w}^T\mathbf{y}_j+b)}{\sqrt{\mathbf{w}^T\Sigma_{\mathbf{x}}\mathbf{w}+\mathbf{w}^T\Sigma_{\mathbf{y}}\mathbf{w}}} \geq \rho\ , \tag{12}$$

[3] This can be directly observed from (6).

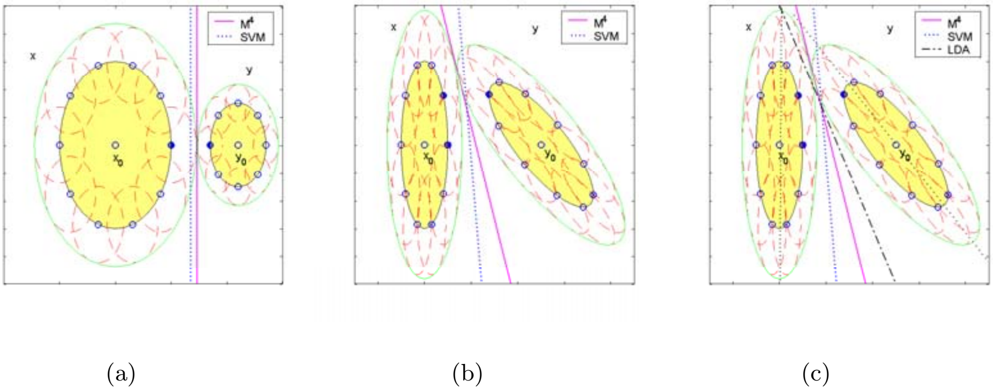


Figure 3. An illustration on the connections between SVM, LDA and M^4. (a) demonstrates SVM omits the data compactness information. (b) demonstrates SVM discards the data orientation information. (c) demonstrates LDA partly yet incompletely considers the data orientation.

where $i = 1, \ldots, N_{\mathbf{x}}$ and $j = 1, \ldots, N_{\mathbf{y}}$. The above optimization is actually a generalized case of LDA, which considers data locally and globally. This is verified as follows.

If one performs the procedure similar to that of Section 2.3.1, the above optimization problem is easily verified to be the following optimization:

$$\max_{\rho, \mathbf{w} \neq \mathbf{0}, b} \quad \rho \quad s.t.$$
$$\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}}) \geq \rho \sqrt{\mathbf{w}^T \Sigma_{\mathbf{x}} \mathbf{w} + \mathbf{w}^T \Sigma_{\mathbf{y}} \mathbf{w}}. \quad (13)$$

One can change (13) 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 ($\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}})$ is implicitly implied as a positive value from (11) and (12)).

Remarks: The extended LDA optimization actually focuses on considering the data orientation, while omitting the data scattering magnitude information. Using the analysis similar to that of Section 2.3.2, we can know that the extended LDA lacks the consideration on the data scattering magnitude. Its decision hyperplane in the example of Figure 3(a) coincides with that of SVM. With respect to the data orientation, it actually uses the average of covariances for two types of data. As illustrated in Figure 3(c), the extended LDA corresponds to the line lying exactly in the middle of the long axes of the $\mathbf{x}$ and $\mathbf{y}$ data. This shows that the extended LDA considers the data orientation partially yet incompletely .

2.4. Nonseparable Case

In this section, we modify the M^4 model to handle the nonseparable case. We need to introduce slack variables in this case. The optimization of M^4 is changed as:

$$\max_{\rho, \mathbf{w} \neq \mathbf{0}, b, \boldsymbol{\xi}} \quad \rho - C \sum_{k=1}^{N_{\mathbf{x}}+N_{\mathbf{y}}} \boldsymbol{\xi}_k \quad s.t. \quad (14)$$
$$(\mathbf{w}^T \mathbf{x}_i + b) \geq \rho \sqrt{\mathbf{w}^T \Sigma_{\mathbf{x}} \mathbf{w}} - \boldsymbol{\xi}_i \ ,$$
$$-(\mathbf{w}^T \mathbf{y}_j + b) \geq \rho \sqrt{\mathbf{w}^T \Sigma_{\mathbf{y}} \mathbf{w}} - \boldsymbol{\xi}_{j+N_{\mathbf{x}}} \ ,$$
$$\boldsymbol{\xi}_k \geq 0 \ ,$$

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. In other words, the above optimization achieves maximizing the minimum margin while minimizing the total training error. The above optimization is easily verified to be an SOCP problem if we fix ρ. We can then update ρ sequently. This is again a sequential SOCP problem and thus can be solved practically.

3. Kernelization

One may note that in the above, the classifier derived from M^4 is provided in a linear configuration. In order to handle nonlinear classification problems, in this section, we seek to use the kernelization trick to map the n-dimensional data points into a high-dimensional feature space $\mathbb{R}^f$, where a linear classifier corresponds to a nonlinear hyperplane in the original space.

The kernel mapping can be formulated as: $\mathbf{x}_i \rightarrow \varphi(\mathbf{x}_i)$, $\mathbf{y}_j \rightarrow \varphi(\mathbf{y}_j)$, where $i = 1, \ldots, N_{\mathbf{x}}$, $j = 1, \ldots, N_{\mathbf{y}}$, and $\varphi : \mathbb{R}^n \rightarrow \mathbb{R}^f$ is a mapping function. The corresponding linear classifier in $\mathbb{R}^f$ is $\boldsymbol{\gamma}^T \varphi(\mathbf{z}) = b$, where $\boldsymbol{\gamma}$, $\varphi(\mathbf{z})$

Table 1. Notations used in Kernelization Theorem of M^4

Notation		
$\mathbf{z} \in \mathbb{R}^{N_\mathbf{x}+N_\mathbf{y}}$	$\mathbf{z}_i :=$	$\mathbf{x}_i \quad i = 1, 2, \ldots, N_\mathbf{x}$.
	$\mathbf{z}_i :=$	$\mathbf{y}_{i-N_\mathbf{x}} \quad i = N_\mathbf{x}+1, N_\mathbf{x}+2, \ldots, N_\mathbf{x}+N_\mathbf{y}$.
$\boldsymbol{\eta} \in \mathbb{R}^{N_\mathbf{x}+N_\mathbf{y}}$	$\boldsymbol{\eta} :=$	$[\mu_1, \ldots, \mu_{N_\mathbf{x}}, \upsilon_1, \ldots, \upsilon_{N_\mathbf{y}}]^T$.
$\mathbf{K}$ is Gram matrix	$\mathbf{K}_{i,j} :=$	$\varphi(\mathbf{z}_i)^T \varphi(\mathbf{z}_j)$.
	$\mathbf{K}_\mathbf{x} :=$	$\begin{pmatrix} \mathbf{K}_{1,1} & \mathbf{K}_{1,2} & \ldots & \mathbf{K}_{1,N_\mathbf{x}+N_\mathbf{y}} \\ \mathbf{K}_{2,1} & \mathbf{K}_{2,2} & \ldots & \mathbf{K}_{2,N_\mathbf{x}+N_\mathbf{y}} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{K}_{N_\mathbf{x},1} & \mathbf{K}_{N_\mathbf{x},2} & \ldots & \mathbf{K}_{N_\mathbf{x},N_\mathbf{x}+N_\mathbf{y}} \end{pmatrix}$.
	$\mathbf{K}_\mathbf{y} :=$	$\begin{pmatrix} \mathbf{K}_{N_\mathbf{x}+1,1} & \mathbf{K}_{N_\mathbf{x}+1,2} & \ldots & \mathbf{K}_{N_\mathbf{x}+1,N_\mathbf{x}+N_\mathbf{y}} \\ \mathbf{K}_{N_\mathbf{x}+2,1} & \mathbf{K}_{N_\mathbf{x}+2,2} & \ldots & \mathbf{K}_{N_\mathbf{x}+2,N_\mathbf{x}+N_\mathbf{y}} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{K}_{N_\mathbf{x}+N_\mathbf{y},1} & \mathbf{K}_{N_\mathbf{x}+N_\mathbf{y},2} & \ldots & \mathbf{K}_{N_\mathbf{x}+N_\mathbf{y},N_\mathbf{x}+N_\mathbf{y}} \end{pmatrix}$.
	$\mathbf{K}_i :=$	$[\mathbf{K}_{i,1}, \mathbf{K}_{i,2}, \ldots, \mathbf{K}_{i,N_\mathbf{x}+N_\mathbf{y}}]^T$.
$\tilde{\mathbf{k}}_\mathbf{x}, \tilde{\mathbf{k}}_\mathbf{y} \in \mathbb{R}^{N_\mathbf{x}+N_\mathbf{y}}$	$[\tilde{\mathbf{k}}_\mathbf{x}]_i :=$	$\frac{1}{N_\mathbf{x}} \sum_{j=1}^{N_\mathbf{x}} \mathbf{K}(\mathbf{x}_j, \mathbf{z}_i)$.
	$[\tilde{\mathbf{k}}_\mathbf{y}]_i :=$	$\frac{1}{N_\mathbf{y}} \sum_{j=1}^{N_\mathbf{y}} \mathbf{K}(\mathbf{y}_j, \mathbf{z}_i)$.
$\mathbf{1}_{N_\mathbf{x}} \in \mathbb{R}^{N_\mathbf{x}}$	$\mathbf{1}_i :=$	$1 \quad i = 1, 2, \ldots N_\mathbf{x}$.
$\mathbf{1}_{N_\mathbf{y}} \in \mathbb{R}^{N_\mathbf{y}}$	$\mathbf{1}_i :=$	$1 \quad i = 1, 2, \ldots N_\mathbf{y}$.
$\tilde{\mathbf{K}} :=$	$\begin{pmatrix} \tilde{\mathbf{K}}_\mathbf{x} \\ \tilde{\mathbf{K}}_\mathbf{y} \end{pmatrix} :=$	$\begin{pmatrix} \mathbf{K}_\mathbf{x} - \mathbf{1}_{N_\mathbf{x}} \tilde{\mathbf{k}}_\mathbf{x}^T \\ \mathbf{K}_\mathbf{y} - \mathbf{1}_{N_\mathbf{y}} \tilde{\mathbf{k}}_\mathbf{y}^T \end{pmatrix}$.

$\in \mathbb{R}^f$, and $b \in \mathbb{R}$.

The optimization of M^4 in the feature space can be written as:

$$\max_{\rho, \boldsymbol{\gamma} \neq \mathbf{0}, b} \quad \rho \quad s.t. \tag{15}$$

$$\frac{(\boldsymbol{\gamma}^T \varphi(\mathbf{x}_i) + b)}{\sqrt{\boldsymbol{\gamma}^T \Sigma_{\varphi(\mathbf{x})} \boldsymbol{\gamma}}} \geq \rho, \quad i = 1, 2, \ldots, N_\mathbf{x} \,, \tag{16}$$

$$\frac{-(\boldsymbol{\gamma}^T \varphi(\mathbf{y}_j) + b)}{\sqrt{\boldsymbol{\gamma}^T \Sigma_{\varphi(\mathbf{y})} \boldsymbol{\gamma}}} \geq \rho, \quad j = 1, 2, \ldots, N_\mathbf{y} \,. \tag{17}$$

However, to make the kernel work, we need to represent the optimization and the final decision hyperplane into a kernel form, $K(\mathbf{z}_1, \mathbf{z}_2) = \varphi(\mathbf{z}_1)^T \varphi(\mathbf{z}_2)$, namely, an inner product form of the mapping data points.

3.1. Kernelization Theory for M^4

In the following, we demonstrate that the kernelization trick indeed works in M^4, provided suitable estimates of means and covariance matrices are applied therein.

Corollary 1 *If the estimates of means and covariance matrices are given in* M^4 *as*

$$\overline{\varphi(\mathbf{x})} = \sum_{i=1}^{N_\mathbf{x}} \varphi(\mathbf{x}_i), \qquad \overline{\varphi(\mathbf{y})} = \sum_{j=1}^{N_\mathbf{y}} \varphi(\mathbf{y}_j)$$

$$\Sigma_{\varphi(\mathbf{x})} = \sum_{i=1}^{N_\mathbf{x}} (\varphi(\mathbf{x}_i) - \overline{\varphi(\mathbf{x})})(\varphi(\mathbf{x}_i) - \overline{\varphi(\mathbf{x})})^T$$

$$\Sigma_{\varphi(\mathbf{y})} = \sum_{j=1}^{N_\mathbf{y}} (\varphi(\mathbf{y}_j) - \overline{\varphi(\mathbf{y})})(\varphi(\mathbf{y}_j) - \overline{\varphi(\mathbf{y})})^T$$

then the optimal $\boldsymbol{\gamma}$ *in (15-17) lies in the space spanned by the training points.*

Proof We write $\boldsymbol{\gamma} = \boldsymbol{\gamma}_p + \boldsymbol{\gamma}_d$, where $\boldsymbol{\gamma}_p$ is the projection of $\boldsymbol{\gamma}$ in the vector space spanned by all the training data points and $\boldsymbol{\gamma}_d$ is the orthogonal component to this span space. By using $\boldsymbol{\gamma}_d^T \varphi(\mathbf{x}_i) = 0$ and $\boldsymbol{\gamma}_d^T \varphi(\mathbf{y}_j) = 0$, one can easily verify that the optimization (15-17) changes to:

$$\max_{\rho, \{\boldsymbol{\gamma}_p, \boldsymbol{\gamma}_d\} \neq \mathbf{0}, b} \quad \rho \quad s.t.$$

$$\frac{-(\boldsymbol{\gamma}_p^T \varphi(\mathbf{x}_i) + b)}{\sqrt{\boldsymbol{\gamma}_p^T \Sigma_{\varphi(\mathbf{x})} \boldsymbol{\gamma}_p + \boldsymbol{\gamma}_d^T \boldsymbol{\gamma}_d}} \geq \rho \,, \tag{18}$$

$$\frac{-(\boldsymbol{\gamma}_p^T \varphi(\mathbf{y}_j) + b)}{\sqrt{\boldsymbol{\gamma}_p^T \Sigma_{\varphi(\mathbf{y})} \boldsymbol{\gamma}_p + \boldsymbol{\gamma}_d^T \boldsymbol{\gamma}_d}} \geq \rho \,, \tag{19}$$

where $i = 1, \ldots, N_\mathbf{x}$, $j = 1, \ldots, N_\mathbf{y}$. Since we intend to maximize the margin ρ, the denominators in the constraints of (18) and (19) need to be as small as possible. This would lead to $\boldsymbol{\gamma}_d = \mathbf{0}$. In other words, the optimal $\boldsymbol{\gamma}$ lies in the vector space spanned by all the training data points. ■

According to Corollary 1, we can write $\boldsymbol{\gamma}$ as $\boldsymbol{\gamma} = \sum_{i=1}^{N_\mathbf{x}} \mu_i \varphi(\mathbf{x}_i) + \sum_{j=1}^{N_\mathbf{y}} \upsilon_j \varphi(\mathbf{y}_j)$, where the coefficients $\mu_i, \upsilon_j \in \mathbb{R}, i = 1, \ldots, N_\mathbf{x}$, $j = 1, \ldots, N_\mathbf{y}$. By simply substituting the above formula into (15-17), we can obtain the kernel form of the optimization of $\mathbf{M}^4$ in the feature space. We present the main result as the following Kernelization Theorem.

Kernelization Theorem of M^4 *The optimal decision hyperplane for* M^4 *involves solving the following* *optimization problem:*

$$
\begin{aligned}
&\max_{\rho,\boldsymbol{\eta}\neq\mathbf{0},b} \quad \rho \quad s.t. \\
&\frac{(\boldsymbol{\eta}^T\mathbf{K}_i+b)}{\sqrt{\frac{1}{N_{\mathbf{x}}}\boldsymbol{\eta}^T\tilde{\mathbf{K}}_{\mathbf{x}}^T\tilde{\mathbf{K}}_{\mathbf{x}}\boldsymbol{\eta}}} \geq \rho, \quad i=1,2,\ldots,N_{\mathbf{x}}\,, \\
&\frac{-(\boldsymbol{\eta}^T\mathbf{K}_{j+N_{\mathbf{x}}}+b)}{\sqrt{\frac{1}{N_{\mathbf{y}}}\boldsymbol{\eta}^T\tilde{\mathbf{K}}_{\mathbf{y}}^T\tilde{\mathbf{K}}_{\mathbf{y}}\boldsymbol{\eta}}} \geq \rho, \quad j=1,2,\ldots,N_{\mathbf{y}}\,.
\end{aligned}
$$

The notations in the above are defined in Table 1.

4. Experiments

In this section, we report the evaluation results. The SOCP problem is solved based on the popular SOCP software Sedumi (Sturm, 1999).

4.1. Evaluations on a Synthetic Toy Data Set

We demonstrate the advantages of our approach in comparison with SVM and MPM in the following synthetic toy data set first.

As illustrated in Figure 4, the data set is generated under two Gaussian distributions: the $\mathbf{x}$ data are randomly sampled from the Gaussian distribution with the mean as $[-3,\ 0]^T$ and the covariance as $[0.5,\ 0; 0,\ 8]$, while the $\mathbf{y}$ data are randomly sampled from another distribution with the mean and the covariance as $[4,\ 0]^T$ and $[6,\ 0; 0,\ 1]$ respectively. Training (test) data, consisting of 20 (60) data points for each class, are presented as o's (+'s) and ×'s (□'s) for $\mathbf{x}$ and $\mathbf{y}$ respectively. Figure 4(a) illustrates the corresponding derived decision hyperplanes from training data, while Figure 4(b) illustrates the performance of these hyerplanes on the test set.

From Figure 4, M^4 achieves the ideal decision boundary, which considers data both locally and globally; whereas SVM obtains the local boundary just in the middle of the support vectors, which discards the global information, namely the statistical "trend" of data occurrence. For MPM, its decision hyperplane is exclusively dependent on the mean and covariance matrices. Thus we can see that this hyperplane coincides with the data shape, i.e., the long axis of training data of $\mathbf{x}$ is nearly in the same direction as the MPM decision hypeplane. However, the estimated mean and covariance may be inaccurate. This results in a relatively lower test accuracy as illustrated in Figure 4(b). In comparison, M^4 incorporates the information of the local points to neutralize the effect caused by inaccurate estimations. The test accuracies are respectively 98.3%, 97.5%, and 95.8% for M^4, SVM, and MPM, which also demonstrates the advantages of M^4.

4.2. Evaluations on Other Data Sets

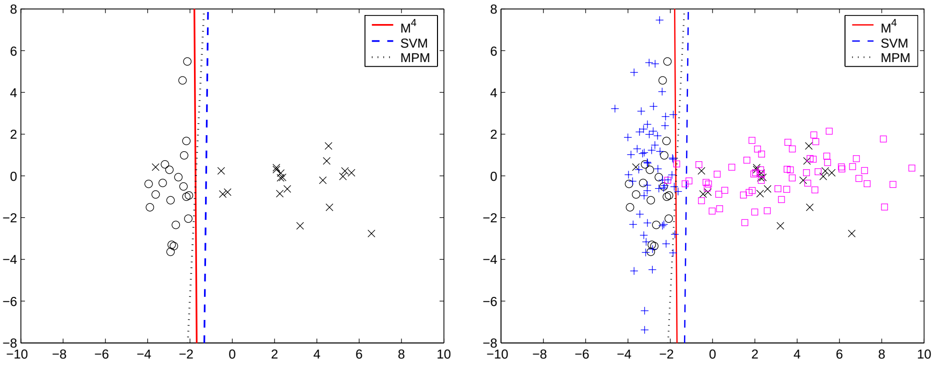
We perform evaluations on seven standard data sets. Data for Twonorm problem were synthetically generated according to (Breiman, 1998). The remaining six data sets were real world data obtained from the UCI machine learning repository. We compared M^4 with SVM and MPM engaging both the linear and Gaussian kernels. The parameter C for both M^4 and SVM was tuned via cross validations, so were the width parameter in the Gaussian kernel for all three models. The final performance results were obtained via the 10−fold cross validation. Table 2 summarizes the evaluation results.

From the results, we observe that M^4 achieves the best overall performance. In comparison with SVM and MPM, M^4 wins five cases in the linear kernel and four cases in the Gaussian kernel. The evaluations on these standard bench-mark data sets demonstrate that it is worth considering data both locally and globally, which is emphasized in M^4. Inspecting the differences between M^4 with SVM, the kernelized M^4 appears marginally better than the kernelized SVM, while the linear M^4 demonstrates a distinctive advantage over the linear SVM. Due to the sparsity of data points in the kernelized space or feature space (compared with the infinite dimension in the Gaussian kernel), this is reasonable, since the plug-in estimation of the covariance matrices may not accurately represent the data information in this case. Further investigations on this topic is highly worthy in the future.

5. Conclusion

We propose a novel large margin classifier, called Maxi-Min Margin Machine. This model learns the decision boundary in both a local and a global fashion. In comparison, other large margin classifiers construct classifiers either locally or globally. For example, a state-of-the-art large margin classifier, Support Vector Machine considers data locally; while another significant model Minimax Probability Machine focuses on building the decision hyperplane exclusively based on the global information. As a critical contribution, we show that M^4 actually presents a unified framework of Support Vector Machine and Minimax Probability Machine. This establishes a bridge between these two important models and provides potentials to exploit the properties of both models in a common way. Moreover, based on our proposed local and global view of data, another popular model, Linear Discriminant Analysis can easily be interpreted and extended as well. The experimental results have also demonstrated the advantages of our new model.

Two important issues are worthy of future investiga-



(a)

(b)

Figure 4. A synthetic toy example to illustrate M^4.

Table 2. Comparisons of classification accuracies among M^4, SVM, and MPM.

Data set	Linear kernel			Gaussian kernel		
	M^4	SVM	MPM	M^4	SVM	MPM
Twonorm(%)	96.5 ± 0.6	95.1 ± 0.7	**97.6 ± 0.5**	96.5 ± 0.7	96.1 ± 0.4	**97.6 ± 0.5**
Breast(%)	**97.5 ± 0.7**	96.6 ± 0.5	96.9 ± 0.8	**97.5 ± 0.6**	96.7 ± 0.4	96.9 ± 0.8
Ionosphere(%)	**87.7 ± 0.8**	86.9 ± 0.6	84.8 ± 0.8	**94.5 ± 0.4**	94.2 ± 0.3	92.3 ± 0.6
Pima(%)	77.7 ± 0.9	**77.9 ± 0.7**	76.1 ± 1.2	77.6 ± 0.8	**78.0 ± 0.5**	76.2 ± 1.2
Sonar(%)	**77.6 ± 1.2**	76.2 ± 1.1	75.5 ± 1.1	84.9 ± 1.2	86.5 ± 1.1	**87.3 ± 0.8**
Vote(%)	**96.1 ± 0.5**	95.1 ± 0.4	94.8 ± 0.4	**96.2 ± 0.5**	95.9 ± 0.6	94.6 ± 0.4
Heart(%)	**86.6 ± 0.8**	84.1 ± 0.7	83.2 ± 0.8	**86.2 ± 0.8**	83.8 ± 0.5	83.1 ± 1.0

tions. First, due to the sparsity of M^4 (with support vectors as well), it would be highly valuable to develop methods to reduce those redundant data points so as to reduce the time complexity of M^4. Moreover, We believe that there is much to gain from both exploiting analogies to SVM and developing specialized optimization procedures for the M^4 model. Second, both SVM and MPM contain a generation error bound. Therefore, exploring the bound of their superset, M^4, is an interesting subject. Third, the current M^4 can only handle binary classifications. How to extent its application into multi-way classifications is also an important topic.

Acknowledgments

The work described in this paper was fully supported by two grants from the Research Grants Council of the Hong Kong SAR, China (Project No. CUHK4182/03E and Project No. CUHK4351/02).

References

Bertsekas, D. P. (1999). *Nonlinear programming*. Belmont, Massashusetts: Athena Scientific. 2nd edition.

Breiman, L. (1998). Arcing classifiers. *Annals of Statistics*, *26(3)*, 801–849.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. Boston, MA: Academic Press.

Lanckriet, G. R. G., Ghaoui, L. E., Bhattacharyya, C., & Jordan, M. I. (2002). A robust minimax approach to classification. *Journal of Machine Learning Research*, *3*, 555–582.

Lobo, M., Vandenberghe, L., , Boyd, S., & Lebret, H. (1998). Applications of second order cone programming. *Linear Algebra and its Applications*, *284*, 193–228.

Pruessner, A. (2003). Conic programming in GAMS. In *Optimization software - the state of the art*. http://www.gamsworld.org/cone/links.htm: INFORMS Atlanta.

Smola, A. J., Bartlett, P. L., Scholkopf, B., & Schuurmans, D. (2000). *Advances in large margin classifiers*. MIT Press.

Sturm, J. (1999). Using Sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, *11*, 625–653.

Vapnik, V. N. (2000). *The nature of statistical learning theory*. New York: Springer. 2nd edition.