Distance Metric Learning with Kernels

Ivor W. Tsang and James T. Kwok
Department of Computer Science
Hong Kong University of Science and Technology
Clear Water Bay
Hong Kong
Email: [ivor,jamesk]@cs.ust.hk

***Abstract*— In this paper, we propose a feature weighting method that works in both the input space and the kernel-induced feature space. It assumes only the availability of similarity (dissimilarity) information, and the number of parameters in the transformation does not depend on the number of features. Besides feature weighting, it can also be regarded as performing nonparametric kernel adaptation. Experimental results on both toy and real-world datasets show promising results.**

I. Introduction

Many classification and clustering algorithms rely on the use of an inner product or distance measure between patterns. Examples include the nearest neighbor classifiers, radial basis function networks, kernel methods and k-means clustering. The commonly-used Euclidean distance metric ($d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (x_i - y_i)^2}$ where $\mathbf{x} = [x_1, \ldots, x_p]', \mathbf{y} = [y_1, \ldots, y_p]' \in \mathbb{R}^p$) assumes that all features are of equal importance, which is often violated in real-world applications. As a remedy, a large number of feature weighting methods have been proposed [1], [11]. Typically, a weight is allocated to each feature, resulting in the diagonally weighted Euclidean distance ($d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (a_i(x_i - y_i))^2} = \sqrt{(\mathbf{x} - \mathbf{y})'\mathbf{A}\mathbf{A}'(\mathbf{x} - \mathbf{y})}$, where a_i is the weight for feature i and $\mathbf{A}$ is a diagonal matrix with $\mathbf{A}_{ii} = a_i$). This, however, still ignores possible correlation among features. An alternative is to use the fully weighted Euclidean distance ($d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})'\mathbf{A}\mathbf{A}'(\mathbf{x} - \mathbf{y})}$ where $\mathbf{A}$ is an arbitrary matrix.), which, however, may lead to an excessive number of free parameters. Various methods have been used to determine $\mathbf{A}$. For example, one can set $\mathbf{A}\mathbf{A}'$ to be the inverse of the covariance matrix, leading to the so-called Mahalanobis distance. Another possibility is to maximize the ratio between intra-class variance and inter-class variance, either globally [4] or locally [6].

A limitation of these feature weighting methods is that they are designed for classification problems, and thus require the availability of class label information. As discussed in [12], it is sometimes easier to obtain similarity (dissimilarity) information instead, i.e., we may only know that certain pairs of patterns are similar (dissimilar). Recently, Xing *et al.* [12] proposed a distance metric learning method that utilizes such information by using convex programming. However, it involves an iterative procedure with projection and eigen decomposition, and can become very costly when the number of features is large.

Another limitation of these methods is that the number of parameters (weights) increases with the number of features. Hence, they cannot be easily kernelized, as the feature space induced by some kernels (such as the Gaussian kernel) can be infinite-dimensional [9]. Note that some feature weighting methods have recently been proposed specifically for support vector machines (SVMs) [5], [10]. However, they again can only select features in the input space, but not in the feature space.

In this paper, we propose a feature weighting method that works in both the input and (kernel-induced) feature spaces. The basic idea is simple and has echoed throughout the literature [3], [4], [12]: similar patterns should be pushed together, dissimilar patterns should be pulled apart. However, as in [12], our proposed method only assumes the availability of similarity (dissimilarity) information. More importantly, as in other kernel methods, the number of parameters is related to the number of patterns, but not to the dimensionality of the patterns. This allows feature weighting in the possibly infinite-dimensional feature space. Alternatively, we can view this method of modifying the distance metric in the feature space as performing nonparametric kernel adaptation.

The rest of this paper is organized as follows. Sections II and III describe the proposed distance metric learning methods in the input and feature space respectively. Experiment results on both toy and real-world datasets are presented in Section IV, and the last section gives some concluding remarks.

II. Input Space Metric Learning

Given any two patterns $\mathbf{x}_i, \mathbf{x}_j$ in the input space $\mathbb{R}^p$, assume that their inner product is defined as $s_{ij} = \mathbf{x}_i'\mathbf{M}\mathbf{x}_j$, where $\mathbf{M}$ is a $p \times p$ positive semi-definite (psd) matrix. Because $\mathbf{M}$ is psd, s_{ij} can also be written as

$$s_{ij} = \mathbf{x}_i'\mathbf{A}\mathbf{A}'\mathbf{x}_j, \tag{1}$$

where $\mathbf{A}$ is a $p \times p$ matrix. The corresponding distance $d_{ij} = d(\mathbf{x}_i, \mathbf{x}_j)$ is then

$$d_{ij}^2 = (\mathbf{x}_i - \mathbf{x}_j)'\mathbf{A}\mathbf{A}'(\mathbf{x}_i - \mathbf{x}_j). \tag{2}$$

The main question is on how to find a suitable transformation $\mathbf{A}$. In the sequel, we will use $\mathbf{M}$ to correspond to the original metric (d), and $\mathbf{A}\mathbf{A}'$ to correspond to the learned metric ($\tilde{d}$).

A. The Primal and Dual Problems

In the following, we denote the sets containing similar and dissimilar pairs by $\mathcal{S}$ and $\mathcal{D}$, which are of sizes $N_{\mathcal{S}}$ and $N_{\mathcal{D}}$ respectively. In order to have improved tightness among similar patterns and better separability for dissimilar patterns, we consider shrinking the distance between similar patterns, while expanding the distance between those that are dissimilar. In other words, when $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}$, $\tilde{d}_{ij}^2$ will be minimized, whereas when $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D}$, $\tilde{d}_{ij}^2 \geq d_{ij}^2$ will be imposed. In the sequel, this expansion will be enforced by requiring that $\tilde{d}_{ij}^2 - d_{ij}^2 \geq \gamma$, where $\gamma \geq 0$ is a variable to be maximized. However, we may not be able to enforce this condition perfectly for all pairs in $\mathcal{D}$ in general. Hence, as for SVMs, slack variables will be introduced in the optimization problem.

Moreover, notice from (1) that the matrix $\mathbf{A}$ basically projects the patterns onto a set of "useful" features. Ideally, this set should be small, meaning that a small rank for $\mathbf{A}$ (or $\mathbf{AA}'$, as $\text{rank}(\mathbf{AA}') = \text{rank}(\mathbf{A})$) will be desirable. Assume that the eigen decomposition of $\mathbf{AA}'$ is $\mathbf{U\Lambda U}'$. Then, $\text{rank}(\mathbf{AA}') = \text{rank}(\mathbf{\Lambda}) = \|\mathbf{\Lambda}\|_0$. However, a direct minimization of this zero norm is difficult and hence we will approximate it by the Euclidean norm $\|\mathbf{\Lambda}\|_2 = \|\mathbf{AA}'\|_2$ in the optimization.

The above discussion leads to the following primal problem, which has a similar form as that of the ν-SVM [8]:

$$\begin{aligned}\min \quad & \tfrac{1}{2}\|\mathbf{AA}'\|^2 + \tfrac{C_{\mathcal{S}}}{N_{\mathcal{S}}}\textstyle\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S} \tilde{d}_{ij}^2 \\ & + C_{\mathcal{D}}\left(-\nu\gamma + \tfrac{1}{N_{\mathcal{D}}}\textstyle\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \xi_{ij}\right),\end{aligned}$$

with respect to the variables $\mathbf{A}, \gamma$ and ξ_{ij}'s, subject to the constraints that $\gamma \geq 0$ and also that for each $(\mathbf{x}_i, \mathbf{x}_j)$ pair in $\mathcal{D}$,

$$\begin{aligned}\tilde{d}_{ij}^2 - d_{ij}^2 &\geq \gamma - \xi_{ij}, \\ \xi_{ij} &\geq 0.\end{aligned}$$

Here, $C_{\mathcal{S}}, C_{\mathcal{D}}, \nu$ are non-negative tunable parameters.

To solve this constrained optimization problem, we use the standard method of Lagrange multipliers. The Lagrangian function is

$$\begin{aligned}\mathcal{L} &= \frac{1}{2}\|\mathbf{AA}'\|^2 + \frac{C_{\mathcal{S}}}{N_{\mathcal{S}}}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} \tilde{d}_{ij}^2 \\ &\quad + C_{\mathcal{D}}\left(-\nu\gamma + \frac{1}{N_{\mathcal{D}}}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \xi_{ij}\right) \\ &\quad - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij}\left(\tilde{d}_{ij}^2 - d_{ij}^2 - \gamma + \xi_{ij}\right) \\ &\quad - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \eta_{ij}\xi_{ij} - \mu\gamma \\ &= \frac{1}{2}\|\mathbf{AA}'\|^2 \\ &\quad + \frac{C_{\mathcal{S}}}{N_{\mathcal{S}}}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} (\mathbf{x}_i - \mathbf{x}_j)'\mathbf{AA}'(\mathbf{x}_i - \mathbf{x}_j) \\ &\quad + C_{\mathcal{D}}\left(-\nu\gamma + \frac{1}{N_{\mathcal{D}}}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \xi_{ij}\right) \\ &\quad - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij}((\mathbf{x}_i - \mathbf{x}_j)'(\mathbf{AA}' - \mathbf{M})(\mathbf{x}_i - \mathbf{x}_j) \\ &\quad - \gamma + \xi_{ij}) - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \eta_{ij}\xi_{ij} - \mu\gamma. \end{aligned} \tag{3}$$

Setting the derivatives with respect to the primal variables to zero, we obtain (on assuming that $\mathbf{A}$ is non-singular[1]),

$$\begin{aligned}\mathbf{AA}' &= \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)' \\ &\quad - \frac{C_{\mathcal{S}}}{N_{\mathcal{S}}}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)', \end{aligned} \tag{4}$$

$$\nu = \frac{1}{C_{\mathcal{D}}}\left(\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij} - \mu\right), \tag{5}$$

$$\alpha_{ij} = \frac{C_{\mathcal{D}}}{N_{\mathcal{D}}} - \eta_{ij}.$$

Substituting into (3), the dual problem becomes

$$\begin{aligned}&\max_{\alpha_{ij}} \textstyle\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij}(\mathbf{x}_i - \mathbf{x}_j)'\mathbf{M}(\mathbf{x}_i - \mathbf{x}_j) \\ &-\tfrac{1}{2}\textstyle\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}}\sum_{(\mathbf{x}_k,\mathbf{x}_l)\in\mathcal{D}} \alpha_{ij}\alpha_{kl}((\mathbf{x}_i - \mathbf{x}_j)'(\mathbf{x}_k - \mathbf{x}_l))^2 \\ &+\tfrac{C_{\mathcal{S}}}{N_{\mathcal{S}}}\textstyle\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}}\sum_{(\mathbf{x}_k,\mathbf{x}_l)\in\mathcal{S}} \alpha_{ij}((\mathbf{x}_i - \mathbf{x}_j)'(\mathbf{x}_k - \mathbf{x}_l))^2,\end{aligned}$$

subject to

$$\left\{\begin{array}{l} \frac{1}{C_{\mathcal{D}}}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij} \geq \nu, \\ \alpha_{ij} \in \left[0, \frac{C_{\mathcal{D}}}{N_{\mathcal{D}}}\right]. \end{array}\right. \tag{6}$$

This is a quadratic programming (QP) problem with $N_{\mathcal{D}}$ variables (and is thus independent of the input dimensionality), and can be solved using standard QP solvers. Moreover, being a QP problem, it does not suffer from the problem of local maximum. Xing *et al.* [12], on the other hand, formulated the

[1]This is always the case in the experiments. Geometrically, this means that the neighborhood at any point will not extend to infinity.

metric learning problem as a convex programming problem. It is also free of local optimum but involves an iterative procedure comprising projection and eigen decomposition. It is thus more costly, especially when the input dimensionality is high. Finally, using (4), we can then obtain the modified inner product from (1) and the corresponding distance metric from (2).

B. "Support Vector" and the Range of ν

The Lagrange multipliers can be studied further by examining the Karush-Kuhn-Tucker (KKT) conditions of the primal problem, which are:

$$\begin{aligned} \alpha_{ij}(\tilde{d}_{ij}^2 - d_{ij}^2 - \gamma + \xi_{ij}) &= 0, \\ \eta_{ij}\xi_{ij} &= 0, \\ \mu\gamma &= 0. \end{aligned} \tag{7}$$

It can be shown that

$$\tilde{d}_{ij}^2 - d_{ij}^2 \begin{cases} = \gamma, & 0 < \alpha_{ij} < \frac{C_{\mathcal{D}}}{N_{\mathcal{D}}}, \\ \geq \gamma, & \alpha_{ij} = 0, \\ \leq \gamma, & \alpha_{ij} = \frac{C_{\mathcal{D}}}{N_{\mathcal{D}}}, \end{cases} \tag{8}$$

which is very similar to that for SVMs [9]. When the Lagrange multipliers are nonzero but less than the upper bound, the constraints $\tilde{d}_{ij}^2 - d_{ij}^2 \geq \gamma$ will be exactly met. When the Lagrange multipliers are zero, the constraints will be met with a larger "margin" and the corresponding patterns will not appear in the final solution in (4) (i.e., they are not "support vectors"). Finally, when the Lagrange multipliers are at the upper bound, the constraints may be violated. The corresponding ξ_{ij}'s may be nonzero and the corresponding pair generates an "error".

The margin γ can be obtained as follows. Consider taking a pair of dissimilar patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ such that the corresponding α_{ij} satisfies $0 < \alpha_{ij} < \frac{C_{\mathcal{D}}}{N_{\mathcal{D}}}$. Using (8), we then have $\gamma = \tilde{d}_{ij}^2 - d_{ij}^2$. In the implementation, we obtain an average value of γ over all $(\mathbf{x}_i, \mathbf{x}_j)$ pairs that satisfy the above criterion.

Moreover, we can obtain from (6),

$$\begin{aligned} \nu &\leq \frac{1}{C_{\mathcal{D}}} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij} = \frac{1}{C_{\mathcal{D}}} \sum_{\alpha_{ij}>0} \alpha_{ij} \\ &\leq \frac{1}{C_{\mathcal{D}}} \sum_{\alpha_{ij}>0} \frac{C_{\mathcal{D}}}{N_{\mathcal{D}}} = \frac{\#(\alpha_{ij}>0)}{N_{\mathcal{D}}}. \end{aligned}$$

Recall that there are $N_{\mathcal{D}}$ α_{ij}'s and all support vectors have $\alpha_{ij} > 0$. Hence, ν is a lower bound on the fraction of support vectors. Moreover, if $\gamma > 0$, then $\mu = 0$ from the KKT condition (7), and (5) yields

$$\begin{aligned} \nu &= \frac{1}{C_{\mathcal{D}}} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij} \geq \frac{1}{C_{\mathcal{D}}} \sum_{\alpha_{ij}=\frac{C_{\mathcal{D}}}{N_{\mathcal{D}}}} \frac{C_{\mathcal{D}}}{N_{\mathcal{D}}} \\ &= \frac{\#(\alpha_{ij} = \frac{C_{\mathcal{D}}}{N_{\mathcal{D}}})}{N_{\mathcal{D}}}. \end{aligned}$$

The inequality holds as the second summation includes only a subset of the nonzero α_{ij}'s. Recall that for the error pairs, α_{ij} is at the upper bound $\frac{C_{\mathcal{D}}}{N_{\mathcal{D}}}$. Hence, ν is also an upper bound on the fraction of errors. These are analogous to the results in [8].

III. Feature Space Metric Learning

For a given kernel function $k(\cdot,\cdot)$, denote the corresponding feature map by ϕ, where $k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)'\phi(\mathbf{x}_j)$. As the results in Section II do not depend explicitly on the input patterns, they can be easily kernelized by replacing $\mathbf{x}$ by $\phi(\mathbf{x})$. For example, the objective function in the dual problem now becomes

$$\begin{aligned} &\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij}(k_{ii} + k_{jj} - 2k_{ij}) \\ &-\frac{1}{2} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \sum_{(\mathbf{x}_k,\mathbf{x}_l)\in\mathcal{D}} \alpha_{ij}\alpha_{kl}(k_{ik} - k_{il} - k_{jk} + k_{jl})^2 \\ &+\frac{C_{\mathcal{S}}}{N_{\mathcal{S}}} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \sum_{(\mathbf{x}_k,\mathbf{x}_l)\in\mathcal{S}} \alpha_{ij}(k_{ik} - k_{il} - k_{jk} + k_{jl})^2, \end{aligned}$$

The modified inner product between $\phi(\mathbf{x}_a), \phi(\mathbf{x}_b)$ effectively leads to a new kernel function $\tilde{k}(\cdot,\cdot)$:

$$\begin{aligned} \tilde{k}(\mathbf{x}_a, \mathbf{x}_b) &= \phi(\mathbf{x}_a)'\mathbf{A}\mathbf{A}'\phi(\mathbf{x}_b) \\ &= \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij}(k_{ai} - k_{aj})(k_{ib} - k_{jb}) \\ &\quad -\frac{C_{\mathcal{S}}}{N_{\mathcal{S}}} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} (k_{ai} - k_{aj})(k_{ib} - k_{jb}). \end{aligned}$$

The corresponding distance metric can also be obtained from (2).

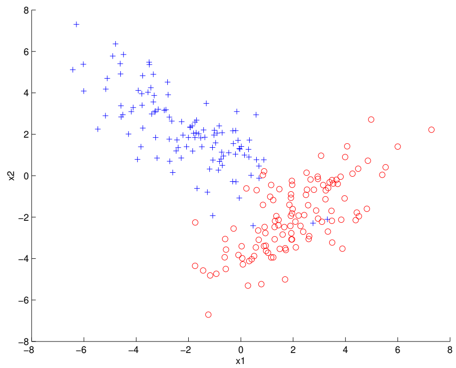
IV. Experiments

In this Section, we perform experiments on a two-class toy dataset and four other real-world datasets[2] (Table I). As mentioned in Section I, the proposed algorithm only assumes the availability of similarity and dissimilarity information. However, to simplify the experiment setups, we will use the class label information as in standard classification problems. The sets $\mathcal{S}$ and $\mathcal{D}$ are constructed by defining two patterns as similar if they belong to the same class, and dissimilar otherwise. 40 patterns are randomly selected to form the training set (for learning both the metric and the classifier), while the remaining patterns are for testing. The proposed method is used to learn the distance metric in both the input space (which corresponds to the linear kernel) and the feature space induced by the Gaussian kernel $\exp(-\beta\|\mathbf{x}_i - \mathbf{x}_j\|^2)$. In the experiments, we use $\frac{1}{\beta} = \frac{1}{N(N-1)}\sum_{i,j=1}^N \|\mathbf{x}_i - \mathbf{x}_j\|^2$, where $\mathbf{x}_1, \ldots, \mathbf{x}_N$ are the training patterns. The 1-nearest neighbor classifier is then employed, using both the original (Euclidean) and the learned metrics, for classification.

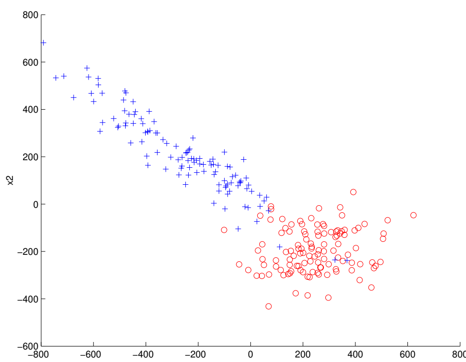
Figure 1 shows the data distributions using the two distance metrics. As can be seen, similar patterns become more

[2] ionosphere, sonar and wine are from the UCI repository [2], and microarray (a DNA microarray dataset for colon cancer) is from http://www.kyb.tuebingen.mpg.de/bs/people/weston/l0.

clustered while dissimilar patterns are more separated. Table II reports the classification accuracies, averaged over 50 repeated trials for each experiment. In general, the learned metric leads to higher accuracies.



(a) Euclidean metric.



(b) Learned metric.

Fig. 1. Data under different distance metrics.

V. Conclusion

In this paper, we propose a feature weighting method that uses only similarity (dissimilarity) information to modify the distance metrics in both the input space and the kernel-induced feature space. As in other kernel methods, the number of parameters in the transformation is related to the number of patterns, but not to the number of features. Besides, this learning method provides a new means for nonparametric kernel adaptation.

Acknowledgments

This research has been partially supported by the Research Grants Council of the Hong Kong Special Administrative Region under grants HKUST2033/00E and HKUST6195/02E.

References

[1] D.W. Aha. Feature weighting for lazy learning algorithms. In H. Liu and H. Motoda, editors, *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Kluwer, Norwell, MA, 1998.

TABLE I

Datasets used in the experiments.

dataset	dim	class	# patterns
toy	2	1	100
		2	100
ionosphere	34	1	126
		2	225
sonar	60	1	111
		2	97
wine	12	1	59
		2	71
		3	48
microarray	2000	1	22
		2	40

TABLE II

1-nearest neighbor classification results.

dataset	kernel	Euclidean metric	learned metric
toy	linear	4.85%	4.80%
	RBF	7.22%	6.81%
ionosphere	linear	20.48%	19.47%
	RBF	18.77%	13.00%
sonar	linear	30.19%	29.78%
	RBF	30.21%	29.48%
wine	linear	30.87%	27.17%
	RBF	31.17%	32.35%
microarray	linear	29.09%	21.00%
	RBF	28.45%	23.82%

[2] C. Blake, E. Keogh, and C.J. Merz. UCI repository of machine learning databases, 1998. http://www.ics.uci.edu/∼mlearn/MLRepository.html University of California, Irvine, Department of Information and Computer Sciences.

[3] C. Domeniconi, J. Peng, and D. Gunopulos. An adaptive metric machine for pattern classification. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, Cambridge, MA, 2001. MIT Press.

[4] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.

[5] Y. Grandvalet and S. Canu. Adaptive scaling for feature selection in SVMs. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

[6] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):607–616, June 1996.

[7] J. Peng, D.R. Heisterkamp, and H.K. Dai. Adaptive kernel metric nearest neighbor classification. In *Proceedings of the International Conference on Pattern Recognition*, Quebec City, Canada, 2002.

[8] B. Schölkopf, A. Smola, R.C. Williamson, and P.L. Bartlett. New support vector algorithms. *Neural Computation*, 12(4):1207–1245, 2000.

[9] B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT, 2002.

[10] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, Cambridge, MA, 2001. MIT Press.

[11] D. Wettschereck, D.W. Aha, and T. Mohri. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, 11:273–314, 1997.

[12] E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.