APPROXIMATE LEAVE-ONE-OUT ERROR ESTIMATION FOR LEARNING WITH SMOOTH, STRICTLY CONVEX MARGIN LOSS FUNCTIONS

Christopher P. Diehl
Applied Physics Laboratory
Johns Hopkins University
11100 Johns Hopkins Road, MS 2-236
Laurel, Maryland 20723
Chris.Diehl@jhuapl.edu

Abstract. Leave-one-out (LOO) error estimation is an important statistical tool for assessing generalization performance. A number of papers have focused on LOO error estimation for support vector machines, but little work has focused on LOO error estimation when learning with smooth, convex margin loss functions. We consider the problem of approximating the LOO error estimate in the context of sparse kernel machine learning. We first motivate a general framework for learning sparse kernel machines that involves minimizing a regularized, smooth, strictly convex margin loss. Then we present an approximation of the LOO error for the family of learning algorithms admissible in the general framework. We examine the implications of the approximation and review preliminary experimental results demonstrating the utility of the approach.

INTRODUCTION

Model selection is a central issue in machine learning. Given multiple models that offer some explanation of the data, the goal of model selection is to identify the model that will yield the best generalization performance. Although significant work has focused on developing tighter bounds on generalization performance [7], practitioners still rely primarily on cross-validation methods to perform model selection.

Leave-one-out (LOO) error estimation is an important statistical tool for assessing generalization performance. The LOO error estimate is known to provide an almost unbiased estimate of the generalization error [13]. A number of papers have focused on LOO error estimation for support vector ma-

chines [20, 9, 18], but little work has focused on LOO error estimation when learning with smooth, convex margin loss functions such as the logistic loss used in kernel logistic regression.

We consider the problem of approximating the LOO error estimate in the context of sparse kernel machine learning. We motivate a general framework for learning sparse kernel machines that involves minimizing a regularized, smooth, strictly convex margin loss. This approach captures the defining attributes of the *reduced support vector machine* (RSVM) proposed by Lee and Mangasarian [11]. In the remainder of the paper, we present an approximation of the LOO error for the family of learning algorithms admissible in the general framework. We examine the implications of the approximation, review preliminary experimental results and present our conclusions.

THE REDUCED SUPPORT VECTOR MACHINE

Constraining the Weight Vector Expansion

To motivate the RSVM approach, we first reflect on properties of the standard L_1-soft margin SVM algorithm. Recall that the objective is to

$$\begin{array}{cc} \min\limits_{\mathbf{w},b} & \frac{1}{2}\|\mathbf{w}\|^2 + \sum\limits_{k=1}^{N} C_k \epsilon_k \\ \text{subject to} & z_k = y_k\left(\mathbf{w}\cdot\Phi(x_k)+b\right) \geq 1-\epsilon_k \\ & \epsilon_k \geq 0 \end{array}$$

where the training set $\mathcal{T} = \{(x_k, y_k) : (x_k, y_k) \in \mathbb{R}^p \times \{\pm 1\}\}$ and the SVM $f(x) = \mathbf{w}\cdot\Phi(x)+b$. From the Karush-Kuhn-Tucker (KKT) conditions for the primal optimization problem introduced above and its corresponding dual, we learn that the weight vector $\mathbf{w} = \sum\limits_{i\in\mathcal{I}_\mathcal{T}} y_i\alpha_i\Phi(x_i)$ where $0 \leq \alpha_i \leq C_k$ and $\mathcal{I}_\mathcal{T}$ is the indexing set for $\mathcal{T}$. The nonzero Lagrange multipliers α_i determine which training examples (support vectors) contribute to the decision boundary.

To explicitly control the number of training examples entering the weight vector expansion, one approach is to simply select a subset $\mathcal{S} \subset \mathcal{T}$ of the training set and assume $\mathbf{w} = \sum\limits_{i\in\mathcal{I}_\mathcal{S}} y_i\alpha_i\Phi(x_i)$. Substituting this expression for $\mathbf{w}$ into the primal and studying the properties of the learning algorithm, we discover two unfortunate outcomes. The constraint on the set of admissible examples $\mathcal{S}$ does not remove the need to solve for N Lagrange multipliers corresponding to the margin constraints. When N is large, the quadratic programming problem is still computationally expensive.

In addition, given there are more Lagrange multipliers than classifier parameters, it is possible that no unique solution exists. In contrast to the result of Burges and Crisp [2], which indicates the SVM solution is unique except in extreme circumstances, it is increasingly likely the solution will not

be unique as the size of the set $\mathcal{S}$ is decreased[1]. Therefore we've removed a desirable property of the original SVM algorithm with this modification. To recover from this, an additional modification is needed.

Smoothing the Margin Loss Function

Consider the class of large margin learning algorithms with objective functions of the form

$$L(\mathbf{w}, b) = \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{N} C_i g(z_i) \tag{1}$$

where $z_i = y_i(\mathbf{w}\cdot\Phi(x_i)+b)$ and g is a convex margin loss. The KKT conditions imply $\mathbf{w} = \sum_{i=1}^{N} -y_i C_i g'(z_i)\Phi(x_i)$ at the optimum [3]. Given the influence of a training example x_i is determined by the slope of the margin loss at the margin value z_i, objective functions with regions of zero slope, such as the hinge loss $g(z) = [1-z]_+$ used in the L_1-soft margin SVM algorithm, induce some level of sparsity. Yet it complicates the learning problem by introducing constraints.

To avoid such complications, Lee and Mangasarian [10, 11] proposed using a smooth, strictly convex margin loss function and removing any constraints on the coefficients in the weight vector expansion. This converts the learning problem into an unconstrained, convex optimization problem that can be solved using a fast Newton method. Given a strictly convex margin loss does not admit sparse solutions, the constraint on the weight vector expansion is key to obtain the level of sparsity we desire.

The RSVM Learning Algorithm

If one views the RSVM approach as being principally defined by the union of the above ideas, one can define a family of RSVM algorithms that differ only in terms of the regularizer and margin loss chosen. Consider the general objective function

$$L(\boldsymbol{\alpha}, b) = \Omega(\boldsymbol{\alpha}, b) + \sum_{i=1}^{N} C_i g(z_i) \tag{2}$$

where $z_i = y_i \left(\sum_{j\in\mathcal{I}_\mathcal{S}} y_j \alpha_j K(x_j, x_i) + b \right)$ and Ω and g are strictly convex. Minimization of this objective function is accomplished using a standard Newton descent algorithm where the updates

$$\begin{bmatrix} \Delta\boldsymbol{\alpha}_k \\ \Delta b_k \end{bmatrix} = -\left(\nabla^2_{\boldsymbol{\alpha},b} L\big|_{(\boldsymbol{\alpha}_{k-1}, b_{k-1})} \right)^{-1} \nabla_{\boldsymbol{\alpha},b} L\big|_{(\boldsymbol{\alpha}_{k-1}, b_{k-1})} \tag{3}$$

[1]In order for a unique solution to exist, the minimum of the constrained optimization problem must be a *regular point* [12]. A regular point exists if the gradient vectors for the constraint equations at the minimum are linearly independent. When the number of active margin constraints exceeds $|\mathcal{S}| + 1$, a regular point does not exist. We omit the details.

are computed at each iteration k. Since we can minimize the primal directly, we gain the benefit of conducting the optimization in a parameter space of dimension $|\mathcal{S}| + 1 \ll N$. In our experiments, the minimization converged in less than 15 iterations.

This framework admits reduced forms of several standard large margin algorithms. In our experiments, we used the regularizer

$$\Omega(\boldsymbol{\alpha}, b) = \frac{1}{2} \sum_{i \in \mathcal{I}_\mathcal{S}} \sum_{j \in \mathcal{I}_\mathcal{S}} y_i y_j \alpha_i \alpha_j K(x_i, x_j) \tag{4}$$

and the margin loss $g(z) = h(1 - z, 1)$ where $h(x, \lambda)$ is the following smooth approximation of $[x]_+$

$$h(x, \lambda) = \frac{1}{\lambda} \log \left(1 + e^{\lambda x}\right), \lambda > 0. \tag{5}$$

This yields an RSVM algorithm similar to the L_1-soft margin SVM. Using the same regularizer with the margin loss $g(z) = h(-z, 1)$ yields an RSVM algorithm for sparse kernel logistic regression. In the formulation presented by Lee and Mangasarian [11], they use the regularizer $\Omega(\boldsymbol{\alpha}, b) = \frac{1}{2}(\boldsymbol{\alpha} \cdot \boldsymbol{\alpha} + b^2)$ and the margin loss $g(z) = h(1 - z, \lambda)^2$ yielding a variant of the L_2-soft margin SVM.

Related Approaches

Clearly the ideas presented above bear strong relation to a number of previous publications. Learning with smooth, convex margin loss functions first gained significant attention as the connection between boosting and large margin algorithms became clear [15, 14]. Since then, several publications [16, 19, 21, 17] have presented related methods that incrementally construct sparse kernel machines by selecting training examples to include in $\mathcal{S}$ that lead to rapid minimization of the objective function. The main difference between the RSVM and these approaches is that the RSVM uses random selection as opposed to active selection.

In [16, 19, 21], model selection was addressed by using a validation set to assess the impact of parameter choices. In [17], a recent generalization bound was used to assess performance. To our knowledge, no one has suggested principled methods for efficient cross-validation in this context. In the next section, we present a method for approximate leave-one-out error estimation for the general framework above. An interesting avenue of future work will be to explore the utility of this approach for active selection algorithms such as [21] that minimize similar objective functions[2].

[2] In [21], the import vector machine algorithm performs sparse kernel logistic regression.

APPROXIMATE LEAVE-ONE-OUT ERROR ESTIMATION

General Approach

By definition, the LOO error estimate is

$$e_{loo} = \frac{1}{N} \sum_{k=1}^{N} \mathcal{I} \left\{ z_k^{(k)} < 0 \right\} \tag{6}$$

where $z_k^{(k)} = y_k f^{(k)}(x_k)$ and $f^{(k)}$ is the classifier resulting from training on all the examples except x_k. Given the exact computation of e_{loo} requires N runs of the learning algorithm, a number of papers have focused on the task of approximating $f^{(k)}$ [20, 8, 9, 18]. [8, 9] present upper bounds on the LOO error while [20, 18] attempt to approximate the result. [20, 9, 18] focus on SVMs while [8] provides a loose upper bound suitable for kernel logistic regression.

Our approach was inspired by insights derived from the study of exact, incremental SVM (un)learning [4, 5] which allows one to exactly compute the LOO error by unlearning each example individually from the SVM solution. Incremental unlearning in this context can be viewed as an incremental perturbation of the SVM solution as the regularization parameter for the unlearned example is reduced to zero. The *span rule* proposed by Vapnik and Chapelle [18] performs a simplified version of the unlearning where it is assumed that none of the training examples change status during the process. This leads to an approximation for $z_k^{(k)}$ that is easily computed in the incremental SVM learning framework.

We will define an approximation for $z_k^{(k)}$ in a similar manner. Recall the general objective function (2) for RSVM learning. To approximate $f^{(k)}$, we first set $C_k = 0$ to remove the influence of x_k, yielding the objective function $L^{(k)}(\boldsymbol{\alpha}, b)$. If x_k is a member of $\mathcal{S}$, we should ideally replace x_k with another randomly selected example to account for the selection process and adapt the regularizer accordingly. We have chosen to keep x_k in the weight vector expansion in such cases.

Next we compute one Newton step with respect to $L^{(k)}(\boldsymbol{\alpha}, b)$ to obtain an estimate of the change in the RSVM parameters

$$\begin{bmatrix} \Delta\boldsymbol{\alpha} \\ \Delta b \end{bmatrix} = - \left(\nabla^2_{\boldsymbol{\alpha},b} L^{(k)} \Big|_{(\boldsymbol{\alpha}_*, b_*)} \right)^{-1} \nabla_{\boldsymbol{\alpha},b} L^{(k)} \Big|_{(\boldsymbol{\alpha}_*, b_*)}. \tag{7}$$

$(\boldsymbol{\alpha}_*, b_*)$ are the classifier parameters that define f. The corresponding estimate for $z_k^{(k)}$ is then

$$z_k^{(k)} \approx z_k + y_k \sum_{j \in \mathcal{I}_\mathcal{S}} y_j \Delta\alpha_j K(x_j, x_k) + y_k \Delta b. \tag{8}$$

Substituting this result into (6), we obtain our estimate of the LOO error.

Efficient Computation of $\Delta \mathbf{z_k}$

The key to efficiently approximating the change in the margin $\Delta z_k = z_k^{(k)} - z_k$ is an update equation relating the inverse Hessians for $f^{(k)}$ and f. One can show that the update equation for the Hessian is

$$\nabla^2_{\boldsymbol{\alpha},b} L^{(k)} = \nabla^2_{\boldsymbol{\alpha},b} L - C_k g''(z_k) \hat{\mathbf{q}}_k \hat{\mathbf{q}}_k^{\mathrm{T}} \tag{9}$$

where $\hat{\mathbf{q}}_k^{\mathrm{T}} = [Q_{jk} \forall j \in \mathcal{I}_{\mathcal{S}} \dots y_k]$ and $Q_{jk} = y_j y_k K(x_j, x_k)$. Given the Hessian update involves subtracting a rank one matrix from the original Hessian, we can use the Sherman-Morrison-Woodbury formula [6] to easily derive the update equation for the inverse Hessian. This yields

$$(\nabla^2_{\boldsymbol{\alpha},b} L^{(k)})^{-1} = (\nabla^2_{\boldsymbol{\alpha},b} L)^{-1} - \frac{\mathbf{v}_k \mathbf{v}_k^{\mathrm{T}}}{\hat{\mathbf{q}}_k^{\mathrm{T}} \mathbf{v}_k - \frac{1}{C_k g''(z_k)}} \tag{10}$$

where $\mathbf{v}_k = (\nabla^2_{\boldsymbol{\alpha},b} L)^{-1} \hat{\mathbf{q}}_k$.

To obtain a simplified expression for Δz_k, we first reexpress (8) in matrix-vector form which yields

$$\Delta z_k \approx -\hat{\mathbf{q}}_k^{\mathrm{T}} \left(\nabla^2_{\boldsymbol{\alpha},b} L^{(k)} \Big|_{(\boldsymbol{\alpha}_*, b_*)} \right)^{-1} \nabla_{\boldsymbol{\alpha},b} L^{(k)} \Big|_{(\boldsymbol{\alpha}_*, b_*)}. \tag{11}$$

The gradient $\nabla_{\boldsymbol{\alpha},b} L^{(k)}$ can be expressed in terms of $\nabla_{\boldsymbol{\alpha},b} L$ as

$$\nabla L^{(k)}_{\boldsymbol{\alpha},b} = \nabla L_{\boldsymbol{\alpha},b} - C_k g'(z_k) \hat{\mathbf{q}}_k. \tag{12}$$

Evaluated at the optimum $(\boldsymbol{\alpha}_*, b_*)$, it reduces to simply

$$\nabla L^{(k)}_{\boldsymbol{\alpha},b} \Big|_{(\boldsymbol{\alpha}_*, b_*)} = -C_k g'(z_k) \hat{\mathbf{q}}_k. \tag{13}$$

Substituting the results from (10) and (13) into (11), we find

$$\Delta z_k \approx C_k g'(z_k) \hat{\mathbf{q}}_k^{\mathrm{T}} \left(\mathbf{H}^{-1} - \frac{\mathbf{v}_k \mathbf{v}_k^{\mathrm{T}}}{\hat{\mathbf{q}}_k^{\mathrm{T}} \mathbf{v}_k - \frac{1}{C_k g''(z_k)}} \right) \hat{\mathbf{q}}_k \tag{14}$$

where $\mathbf{H}^{-1} = \left(\nabla^2_{\boldsymbol{\alpha},b} L \Big|_{(\boldsymbol{\alpha}_*, b_*)} \right)^{-1}$ and $\mathbf{v}_k = \mathbf{H}^{-1} \hat{\mathbf{q}}_k$. After a bit of algebra, this simplifies to

$$\Delta z_k \approx \frac{C_k g'(z_k) \hat{\mathbf{q}}_k^{\mathrm{T}} \mathbf{H}^{-1} \hat{\mathbf{q}}_k}{1 - C_k g''(z_k) \hat{\mathbf{q}}_k^{\mathrm{T}} \mathbf{H}^{-1} \hat{\mathbf{q}}_k}. \tag{15}$$

After computing $\hat{\mathbf{q}}_k^{\mathrm{T}} \mathbf{H}^{-1} \hat{\mathbf{q}}_k$, the computation of the approximation is trivial.

Margin Sensitivity and Unlearning

As a given example x_k is unlearned by reducing the regularization parameter C_k to zero, we expect that the corresponding margin z_k will decrease monotonically. If this is true, the margin sensitivity $\frac{\partial z_k}{\partial C_k}$ should always be greater than or equal to zero. (15) provides an approximation for Δz_k when $\Delta C_k = -C_k$. Generalizing this result, we obtain an approximation for $\frac{\Delta z_k}{\Delta C_k}$ given an arbitrary ΔC_k

$$\frac{\Delta z_k}{\Delta C_k} \approx \frac{-g'(z_k)\hat{\mathbf{q}}_k^{\mathrm{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k}{1 + \Delta C_k g''(z_k)\hat{\mathbf{q}}_k^{\mathrm{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k}. \tag{16}$$

Taking the limit as $\Delta C_k \to 0$, we obtain the margin sensitivity $\frac{\partial z_k}{\partial C_k}$

$$\frac{\partial z_k}{\partial C_k} = -g'(z_k)\hat{\mathbf{q}}_k^{\mathrm{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k. \tag{17}$$

Consider the signs of the components of (17). Given $\mathbf{H}^{-1}$ is positive definite, $\hat{\mathbf{q}}_k^{\mathrm{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k \geq 0$. By definition, $g'(z_k) \leq 0$ for all convex margin losses. Therefore $\frac{\partial z_k}{\partial C_k} \geq 0$ as expected.

This result has two implications for the LOO process. For all examples with negative margins prior to unlearning, there is no need to estimate Δz_k since the margin will only become more negative. Therefore x_k will continue to be misclassified and can be counted as an error immediately. This result also warns us to be wary of the margin approximation if the sign of Δz_k is positive. If ΔC_k is large enough such that $1 + \Delta C_k g''(z_k)\hat{\mathbf{q}}_k^{\mathrm{T}}\mathbf{H}^{-1}\hat{\mathbf{q}}_k < 0$, the approximation is surely breaking down.

RESULTS AND CONCLUSIONS

We conducted a preliminary series of experiments to test the model selection utility of the LOO approximation. We investigated this by collecting statistics on the deviation between the test error for the classifier f_{loo} selected using the LOO error and the test error for the best performing classifier f^*. An experiment on a given dataset consisted of 30 trials. Each trial involved randomly selecting $|\mathcal{S}|$ examples to include in the weight vector expansion. During each trial, the kernel was fixed and the regularization parameter was incremented over a given range. The RBF kernel $K(x, y) = e^{\frac{-\|x-y\|^2}{\sigma^2}}$ was used in all of the experiments. The tests were conducted on four datasets: the Checkerboard dataset and the Tic-Tac-Toe, Pima Indians and German datasets from the UCI repository [1].

Table 1 presents summary statistics for the experiments. The means and standard deviations of the difference between the test errors for f^* and f_{loo} are shown. To provide a sense of the significance of these deviations, the 95% confidence intervals for test sets of the specified sizes are given. We used the Hoeffding inequality [7] to compute the confidence bounds.

Table 1: Test Error Deviation Statistics

Dataset	# *Train*	# *Test*	$\lvert\mathcal{S}\rvert$	*Mean Test Error* Δ	*Std. Dev. Test Error* Δ	95% *Confidence Interval* $\pm\epsilon$
Checkerboard	1000	8000	50	0.00062	0.0007	0.015
Tic-Tac-Toe	479	479	40	0.0026	0.0033	0.062
Pima Indians	384	384	30	0.0065	0.0088	0.069
German	500	500	50	0.011	0.011	0.061

The performance of the model selection procedure on the Checkerboard and Tic-Tac-Toe datasets was excellent. The LOO error consistently identified classifiers yielding test error rates at or near the optimal rates. Performance on the Pima Indians and German datasets was good over most trials. Yet there were several trials with more significant test error deviations that warranted further investigation.

Figure 1 shows a series of plots of the LOO error and the test error for classifiers with a range of regularization parameters. The first four plots (a)-(d) provide a look at one randomly selected trial for each dataset. In all cases, a classifier with near optimal or optimal performance was selected.

The remaining plots (e)-(f) illustrate two known failure modes that occur when using the LOO error for model selection. In the first case, multiple minima in the LOO error curve introduce ambiguity in the model selection process. Without additional constraints, there is no clear choice. In the second case, the LOO error fails to adequately track the test error. Both the exact and approximate LOO error curves provide misleading guidance for model selection.

In future work, we will compare the exact and approximate LOO error through extensive experimentation to fully characterize the performance of the approximation. Then we intend to define an approximation of the LOO error variance based on the ideas presented here. Such an estimate will be useful for mitigating the failure modes outlined above. Meanwhile we will conduct additional experiments to investigate why the LOO error does not track the test error appropriately in certain cases. Understanding this failure mode is critical in order to devise improved model selection approaches.

REFERENCES

[1] C. Blake and C. Merz, "UCI Repository of machine learning databases," 1998, University of California, Irvine, Dept. of Information and Computer Sciences, http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[2] C. J. C. Burges and D. J. Crisp, "Uniqueness of the SVM Solution," in S. A. Solla, T. K. Leen and K.-R. Müller (eds.), **Advances in Neural Information Processing Systems 12**, Morgan Kaufmann, 2000.

[3] G. Cauwenberghs, "Kernel Machine Learning: A Systems Perspective," ISCAS Tutorial on Statistical Learning Theory and Support Vector Machines, http://bach.ece.jhu.edu/svm/iscas2001/iscas2001.pdf, 2001.

[4] G. Cauwenberghs and T. Poggio, "Incremental and Decremental Support Vector Machine Learning," in **Advances in Neural Information Processing**

Systems 13, MIT Press, 2001.

[5] C. P. Diehl and G. Cauwenberghs, "SVM Incremental Learning, Adaptation and Optimization," in **Proceedings, International Joint Conference on Neural Networks**, 2003, pp. 2685–2690.

[6] G. H. Golub and C. F. V. Loan, **Matrix Computations**, The Johns Hopkins University Press, 1996.

[7] R. Herbrich, **Learning Kernel Classifiers**, MIT Press, 2002.

[8] T. Jaakkola and D. Haussler, "Probabilistic Kernel Regression Models," in **Proceedings of the Conference on AI and Statistics**, 1999.

[9] T. Joachims, "Estimating the Generalization Performance of an SVM Efficiently," in **Proceedings, International Conference on Machine Learning (ICML)**, Morgan Kaufman, 2000.

[10] Y.-J. Lee and O. Mangasarian, "SSVM: A Smooth Support Vector Machine for Classification," **Computational Optimization and Applications**, vol. 20, no. 1, pp. 5–22, October 2001.

[11] Y.-J. Lee and O. L. Mangasarian, "RSVM: Reduced Support Vector Machines," in **CD Proceedings of the SIAM International Conference on Data Mining**, SIAM, April 2001.

[12] D. G. Luenberger, **Linear and Nonlinear Programming**, Addison Wesley, 1989.

[13] A. Luntz and V. Brailovsky, "On Estimation of Characters Obtained in Statistical Procedure of Recognition," **Technicheskaya Kibernetica**, vol. 3, 1969.

[14] L. Mason, J. Baxter, P. Bartlett and M. Frean, "Functional Gradient Techniques for Combining Hypotheses," in A. Smola, P. Bartlett, B. Schölkopf and D. Schuurmans (eds.), **Advances in Large Margin Classifiers**, MIT Press, 1999.

[15] R. E. Schapire, Y. Freund, P. Bartlett and W. S. Lee, "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods," in **Proceedings, Fourteenth International Conference on Machine Learning**, 1997, pp. 322–330.

[16] Y. Singer, "Leveraged Vector Machines," in S. A. Solla, T. K. Leen and K.-R. Müller (eds.), **Advances in Neural Information Processing Systems 12**, Morgan Kaufmann, 2000.

[17] T. R. Strohmann, A. Belitski, G. Z. Grudic and D. DeCoste, "Sparse Greedy Minimax Probability Machine Classification," in **Advances in Neural Information Processing Systems 16**, Cambridge, MA: MIT Press, 2004.

[18] V. V. Vapnik and O. Chapelle, "Bounds on Error Expectation for Support Vector Machines," **Neural Computation**, vol. 12, no. 9, 2000.

[19] P. Vincent and Y. Bengio, "Kernel Matching Pursuit," **Machine Learning**, vol. 48, no. 1-3, pp. 165–187, 2002.

[20] G. Wahba, "Support Vector Machines, Reproducing Kernel Hilbert Spaces and the Randomized GACV," in **Advances in Kernel Methods– Support Vector Learning**, Cambridge MA: MIT Press, 1998.

[21] J. Zhu and T. Hastie, "Kernel Logistic Regression and the Import Vector Machine," in T. G. Dietterich, S. Becker and Z. Ghahramani (eds.), **Advances in Neural Information Processing Systems 14**, Cambridge, MA: MIT Press, 2002.

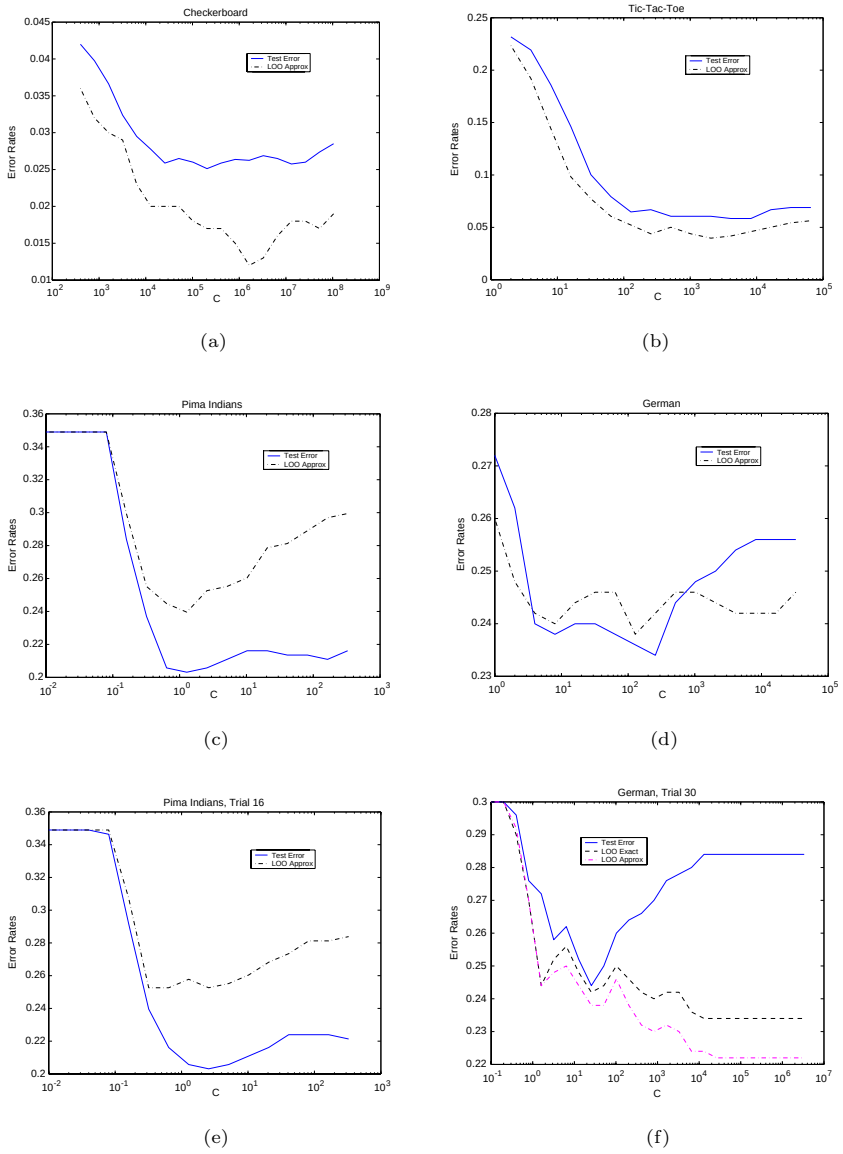

Figure 1: LOO and test error rates: (a) Checkerboard: 50 kernels (b) Tic-Tac-Toe: 40 kernels (c) Pima Indians: 30 kernels (d) German: 50 kernels ; Conditions leading to poor model selection performance: (e) Multiple minima lead to model selection ambiguity (f) LOO error curve fails to adequately track the test error.