Optimization Methods and Software
Vol. 00, No. 00, Month 200x, 1–20

Classification model selection via bilevel programming

G. KUNAPULI*†, K. P. BENNETT†, JING HU† and JONG-SHI PANG‡

†Department of Mathematical Sciences,
Rensselaer Polytechnic Institute, 110 8th Street, Troy NY 12180.
‡ Department of Industrial and Enterprise Systems Engineering,
University of Illinois at Urbana-Champaign, 104 S. Mathews Ave., Urbana IL 61801.

(Received 31 July 2006; revised 24 January 2007; in final form 23 October 2007)

Support vector machines and related classification models require the solution of convex optimization problems that have one or more regularization hyper-parameters. Typically, the hyper-parameters are selected to minimize cross validated estimates of the out-of-sample classification error of the model. This cross-validation optimization problem can be formulated as a bilevel program in which the outer-level objective minimizes the average number of misclassified points across the cross-validation folds, subject to inner-level constraints such that the classification functions for each fold are (exactly or nearly) optimal for the selected hyper-parameters. Feature selection is included in the bilevel program in the form of bound constraints in the weights. The resulting bilevel problem is converted to a mathematical program with linear equilibrium constraints, which is solved using state-of-the-art optimization methods. This approach is significantly more versatile than commonly used grid search procedures, enabling, in particular, the use of models with many hyper-parameters. Numerical results demonstrate the practicality of this approach for model selection in machine learning.

Keywords: Support vector classification; Cross validation; Bilevel Programming; Model Selection; Feature Selection

1 Introduction

Support Vector Machines (SVM) [1,2] are one of the most widely used methods for classification. The underlying quadratic programming problem is convex (thus, is generally not difficult to deal with, both theoretically and computationally), but typically it contains *hyper-parameters* that must be selected by the users. Despite many interesting attempts to use bounds or other techniques [3–5], the most widely accepted and commonly used method for select-

∗Corresponding author. Email: kunapg@rpi.edu

ing these hyper-parameters is still based on minimizing cross-validated estimates of the classification errors. For example, one might define a grid over the hyper-parameters of interest, and then perform 10-fold cross validation for each of the grid values. Since the size of the grid grows exponentially with the number of hyper-parameters, grid search becomes prohibitively expensive for a large number of parameters. In this work, we use bilevel programming to identify hyper-parameters for classification problems.

The work nontrivially extends our recent work on applying the methodology of bilevel programming to parameter identification problems in machine learning. The motivation and benefits of the bilevel approach are explained in our previous paper, [6], where we have discussed the application to constrained regression within the framework of cross validation. In essence, unlike many traditional grid search methods used in machine learning that are severely restricted by the number of hyper-parameters to be searched, the bilevel approach enables the identification of many such parameters all at once by way of the state-of-the-art optimization methods and their softwares (such as those publicly available on the NEOS servers). Another important advantage of the bilevel approach is its modeling versatility in handling multiple machine learning goals simultaneously and efficiently; these include optimal choice of model parameters [3, 7], feature selection for dimension reduction [8], inexact cross validation, kernelization to handle nonlinear data sets [9], and variance control for fold consistency through multi-tasking.

Solution path methods [5, 10] also tackle selection of regularization parameters as a continuous optimization problem using parametric linear/quadratic programming, which necessarily restricts the selection to one single parameter only. These methods solve the problem of picking the best parameter using one training set and one testing set very efficiently. One paper used this approach to optimize parameters [11] by combining the solution paths of the two paths for each of the two parameters. Solution path approaches are a special case of the bilevel approach, but the bilevel approach is far more general in that it can be applied to many parameters and alternative measures of generalization, based on many solutions such as cross-validation.

From the optimization point of view, bilevel programs resulting from these applications belong to the general class of mathematical programs with equilibrium constraints (MPECs), [12], for which there are extensive advances in theory, algorithms, and software in recent years. Some selected references focusing on algorithm developments and analysis include [13–25].

We focus on the bilevel binary classification problem where the main task is to classify data into two groups according to a linear model using a classical support-vector (SV) classifier [1]. The hyper-parameters are selected to minimize the T-fold cross validated estimate of the out-of-sample misclassification error. Each fold of training defines an inner-level convex quadratic

program (QP) with parameters constrained by some bounds that are part of the overall variables to be optimized; such bounds provide a mechanism for feature selection whereby those features corresponding to small bounds in the solution of the bilevel problem will be deemed insignificant. The outer-level problem minimizes the T-fold average classification error based on the optimal solutions of the inner-level QPs for all possible hyper-parameters. Using the approach in [26], we add inner-level linear programs to compute the number of misclassified test points for each fold. In principle, the objective functions in the inner-level classification optimization problems could be rather general; the only restriction we impose is their convexity so that the only non-convexity generated by the inner-level problems in the MPEC is essentially the complementarity slackness in the optimality conditions of the inner-level problems.

The organization of the paper is as follows. In Section 2, we present the mathematical formulation of the bilevel cross validation classification model and show how it can be converted to an instance of an MPEC. In Section 3, we illustrate the power of the formulations to address variations of the classification problems. In Section 4, we describe grid search and present a relaxed nonlinear programming reformulation of the MPEC called inexact cross validation. In Section 5, we describe the experimental setup, the data sets used, and present some computational results comparing the grid search and bilevel cross validation methods. We conclude the paper with some final remarks in Section 6.

2 Model Formulation

We first say a few words about our notation. Let Ω denote a given finite set of $\ell = |\Omega|$ labeled data points, $\{(\mathbf{x}_i, y_i)\}_{i=1}^{\ell} \subset \mathbb{R}^{n+1}$. Since we are interested in the binary classification case, the labels y_i are ± 1. Let the set of indices for the points in Ω be $\mathcal{N} = \{1, \ldots, \ell\}$. For T-fold cross validation, Ω is partitioned into T pairwise disjoint subsets, Ω_t, called the *validation sets*. The sets $\overline{\Omega}_t = \Omega \setminus \Omega_t$ are the *training sets* within each fold. The corresponding index sets for the validation and training sets are $\mathcal{N}_t$ and $\overline{\mathcal{N}}_t$ respectively. The hyperplane trained within the t-th fold using the training set $\overline{\Omega}_t$ is identified by the pair $(\mathbf{w}^t, b_t) \in \mathbb{R}^{n+1}$. For compactness of notation, the vectors $\mathbf{w}^t$ are collected, column-wise, into the matrix $\mathbf{W} \in \mathbb{R}^{n \times T}$, and the scalars b_t into the vector $\mathbf{b} \in \mathbb{R}^T$. A vector of ones of arbitrary dimension is denoted by $\mathbb{1}$. Given two vectors, $\mathbf{r}$ and $\mathbf{s} \in \mathbb{R}^n$, the complementarity condition $\mathbf{r} \perp \mathbf{s}$ means $\mathbf{r}'\mathbf{s} = 0$, where the prime $'$ denotes the transpose; and $\mathbf{r}_\star$ denotes the step function applied to each component of the vector $\mathbf{r}$ as defined in (7).

The well-known SV classification problem depends on a nonnegative regularization parameter, λ, which is selected through cross validation based on

the average misclassification error measured on the out-of-sample data, i.e, the validation sets. Specifically, the basic bilevel model for support vector classification, is formulated as follows:

$$\begin{array}{ll} \underset{\mathbf{W},\mathbf{b},\lambda,\overline{\mathbf{w}}}{\text{minimize}} & \Theta(\mathbf{W},\mathbf{b}) \\ \text{subject to} & \lambda_{\mathbf{lb}} \leq \lambda \leq \lambda_{\mathbf{ub}}, \quad \overline{\mathbf{w}}_{\mathbf{lb}} \leq \overline{\mathbf{w}} \leq \overline{\mathbf{w}}_{\mathbf{ub}}, \\ & \text{and for } t = 1,\ldots,T, \end{array} \tag{1}$$

$$(\mathbf{w}^t, b_t) \in \underset{\substack{-\overline{\mathbf{w}} \leq \mathbf{w} \leq \overline{\mathbf{w}} \\ b \in \mathbb{R}}}{\arg\min} \left\{ \frac{\lambda}{2} \| \mathbf{w} \|_2^2 + \sum_{j \in \overline{\mathcal{N}}_t} \max\left(1 - y_j(\mathbf{x}_j'\mathbf{w} - b), 0\right) \right\}. \tag{2}$$

Problem 1 is called the outer-level problem and subproblems 2 are called the inner-level problems. The outer-level objective, $\Theta(\mathbf{W},\mathbf{b})$, is some measure of validation accuracy over all the folds, typically the average number of misclassifications. There are T inner-level subproblems, one for each fold in T-fold cross validation. The arg min is the last constraint in (1) and denotes the set of all optimal solutions to the T convex optimization problems (2). Each t-th subproblem is simply a classical support vector classification problem applied to the corresponding training set, $\overline{\Omega}_t$, along with the additional box constraint of the form $-\overline{\mathbf{w}} \leq \mathbf{w} \leq \overline{\mathbf{w}}$, where $\overline{\mathbf{w}}$ is a variable in the overall bilevel optimization; in turn, $\overline{\mathbf{w}}$ is restricted to given bounds $\overline{\mathbf{w}}_{\mathbf{ub}} \geq \overline{\mathbf{w}}_{\mathbf{lb}} \geq 0$.

The symmetric box constraint is included for the purposes of wrapper-type feature selection and regularization. In addition, the box constraint was selected to illustrate that the bilevel approach can successfully optimize many hyper-parameters. Note that there is one box constraint parameter for every descriptor. Consider a particular feature that is expected to be redundant or irrelevant i.e., that feature does not contribute much to the final classifier. Then, the corresponding weight in $\overline{\mathbf{w}}$ would be small or zero, which, in turn, constrains the corresponding weights in each $\mathbf{w}^t$, thereby effectively controlling their capacity and potentially increasing generalization performance. The bilevel program will effectively be a wrapper feature selection method. Wrapper methods search for subsets of feature that optimize estimates of the testing generalization error for models trained with those features [27]. Sparse 1-norm regularization can also be used for feature selection but the subset features in each of the CV folds illustrates great variability [8]. The box constraints will ensure that a consistent subset of the features will be used across all the folds. If $\overline{\mathbf{w}}$ can be picked effectively, the box constrained SVM, (2), could represent a fundamentally new way to perform feature selection. Thus, the box constraints are embedded in the bilevel cross validation scheme and $\overline{\mathbf{w}}$ becomes a vector of hyper-parameters in the problem.

Note that we use $\frac{\lambda}{2}\|\mathbf{w}\|_2^2$ for regularization rather than the typical term

$C \sum_{j \in \overline{\mathcal{N}}_t} \max(1 - y_j(\mathbf{x}_j' \mathbf{w} - b), 0)$, where C is the parameter to be chosen. This is due to our empirical observation that the former is more numerically robust than the latter within the bilevel setting. Clearly, the two modes of regularization are equivalent for each inner-level problem with $C = \frac{1}{\lambda}$, provided that both parameters are positive. Note that the bilevel program selects the hyperparameters. The final classifier can be constructed by optimizing a single instance of the lower-level problem using the optimal hyper-parameters and all of the training data. In this case, the final λ should be scaled by $\frac{T}{T-1}$ to account for the larger training set size.

Similar to $\overline{\mathbf{w}}$, the parameter λ is subject to given bounds $\lambda_{\mathbf{ub}} \geq \lambda_{\mathbf{lb}} > 0$. This is done for three reasons: first, to facilitate direct comparison with grid search based cross validation (see Section 4.2); second, to improve the stability and speed up convergence of a general purpose NLP solver; and third, to ensure the positivity of the parameters λ and $\overline{\mathbf{w}}$: so that the bilevel approach yields, in case of the former, a nonzero regularization parameter, in case of the latter, a nontrivial box constraint for feature selection.

2.1 *The inner-level problems*

As mentioned above, there are T inner-level problems that model the training of classifiers within each fold. Consider the inner-level problem corresponding to the t-th fold i.e., the t-th training set, $\overline{\Omega}_t$, indexed by $\overline{\mathcal{N}}_t$, is used. With λ and $\overline{\mathbf{w}}$ fixed in this subproblem, we introduce slack variables, $\boldsymbol{\xi}^t$, in (2) to reformulate the max function using standard linear programming techniques. This gives the the box-constrained SV classifier (BoxSVC) which is nearly identical to the classical SVM for classification, except that it has the additional box constraint for regularization and feature selection:

$$\begin{aligned} \underset{\mathbf{w}^t, b_t, \boldsymbol{\xi}^t}{\text{minimize}} \quad & \frac{\lambda}{2} \|\mathbf{w}^t\|_2^2 + \sum_{j \in \overline{\mathcal{N}}_t} \xi_j^t \\ \text{subject to } & -\overline{\mathbf{w}} \leq \mathbf{w}^t \leq \overline{\mathbf{w}}, \\ & \left\{ \begin{array}{l} y_j(\mathbf{x}_j' \mathbf{w}^t - b_t) \geq 1 - \xi_j^t \\ \xi_j^t \geq 0 \end{array} \right\} \forall\, j \in \overline{\mathcal{N}}_t. \end{aligned} \tag{3}$$

The BoxSVC is a convex quadratic program in the variables $\mathbf{w}^t$, b_t and $\{\xi_j^t\}_{j \in \overline{\mathcal{N}}_t}$. Let $\boldsymbol{\gamma}^{t,-}$ and $\boldsymbol{\gamma}^{t,+}$ be the multipliers of the lower and upper bound constraints $-\overline{\mathbf{w}} \leq \mathbf{w}^t \leq \overline{\mathbf{w}}$ respectively and α_j^t be the multiplier for the hyperplane constraint, $y_j(\mathbf{x}_j' \mathbf{w}^t - b_t) \geq 1 - \xi_j^t$. Using these multipliers, we can write down the primal and dual feasibility and complementarity slackness conditions

of (3) compactly as follows:

$$\left.\begin{aligned} 0 \le \alpha_j^t \quad &\perp y_j(\mathbf{x}_j'\mathbf{w}^t - b_t) - 1 + \xi_j^t \ge 0 \\ 0 \le \xi_j^t \quad &\perp 1 - \alpha_j^t \ge 0 \end{aligned}\right\} \forall\, j \in \overline{\mathcal{N}}_t, \\ 0 \le \boldsymbol{\gamma}^{t,+} \perp \overline{\mathbf{w}} - \mathbf{w}^t \ge 0, \\ 0 \le \boldsymbol{\gamma}^{t,-} \perp \overline{\mathbf{w}} + \mathbf{w}^t \ge 0, \tag{4}$$

which together with the following first-order conditions,

$$\lambda\, \mathbf{w}^t - \sum_{j \in \overline{\mathcal{N}}_t} y_j \alpha_j^t \mathbf{x}_j + \boldsymbol{\gamma}^{t,+} - \boldsymbol{\gamma}^{t,-} = 0, \\ \sum_{j \in \overline{\mathcal{N}}_t} y_j \alpha_j^t = 0, \tag{5}$$

constitute the Karush-Kuhn-Tucker (KKT) optimality conditions to (3). The KKT conditions are necessary and sufficient conditions for the optimal solution of (3). Thus the inner-level optimization problems (2) can be replaced with the system of equations (4) and (5).

2.2 *The outer-level optimization*

The inner-level problems solve T box-constrained SV classification problems on the training sets to yield T hyperplanes, $(\mathbf{w}^t, b_t)$, one for each fold. The outer-level objective function is a measure of generalization error based on the T out-of-sample validation sets, which we minimize. The measure used here is the classical cross-validation error for classification, the average number of points misclassified. The outer-level objective that achieves this can be written using the step function, $()_\star$, as

$$\Theta(\mathbf{W}, \mathbf{b}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{|\,\mathcal{N}_t\,|} \sum_{i \in \mathcal{N}_t} \left[-y_i(\mathbf{x}_i'\mathbf{w}^t - b_t)\right]_\star . \tag{6}$$

Note that in the inner summation, Ω_t, the t-th validation set, indexed by $\mathcal{N}_t$, is used. The inner summation averages the number of misclassifications within each fold while the outer summation averages the averaged misclassification error over the folds. The step function used in (6) can be defined, componen-

twise, for a vector, $\mathbf{r}$, as

$$(\mathbf{r}_\star)_i = \begin{cases} 1, & \text{if } r_i > 0, \\ 0, & \text{if } r_i \leq 0. \end{cases} \tag{7}$$

It is clear that $()_\star$ is discontinuous and that (6) cannot be used directly in the bilevel setting. The step function, however, can be characterized as the solution to a linear program as demonstrated in [26], i.e.,

$$\mathbf{r}_\star = \arg\min_{\boldsymbol{\zeta}} \{-\boldsymbol{\zeta}'\mathbf{r} : 0 \leq \boldsymbol{\zeta} \leq \mathbb{1}\}. \tag{8}$$

Thus, we have to solve T linear programs of the form (9) to determine which validation points, $\mathbf{x}_i \in \mathcal{N}_t$, are misclassified within the t-th fold, i.e., when the sign of $y_i(\mathbf{x}_i'\mathbf{w}^t - b_t)$ is *negative*. These LPs are inserted as inner-level problems into the bilevel setting in order to recast the discontinuous outer-level objective into a continuous one. They yield $\boldsymbol{\zeta}^t = \left[-y_i(\mathbf{x}_i'\mathbf{w}^t - b_t)\right]_\star$, with $\zeta_i^t = 1$ if the point $\mathbf{x}_i$ is misclassified and 0 otherwise. Finally, it should be noted that if $\mathbf{x}_i$ lies on the hyperplane, $(\mathbf{w}^t, b_t)$, then we will have $0 < \zeta_i^t < 1$.

$$\boldsymbol{\zeta}^t \in \arg\min_{0 \leq \boldsymbol{\zeta} \leq \mathbb{1}} \left\{ \sum_{i \in \mathcal{N}_t} \zeta_i y_i \left(\mathbf{x}_i'\mathbf{w}^t - b_t\right) \right\}. \tag{9}$$

Returning to the general case, we introduce additional multipliers, $\mathbf{z}$, for the constraint $\boldsymbol{\zeta} \leq \mathbb{1}$. Consequently, any solution to (8) should satisfy the following linear complementarity conditions:

$$\begin{aligned} 0 \leq \boldsymbol{\zeta} \perp -\mathbf{r} + \mathbf{z} \geq 0, \\ 0 \leq \mathbf{z} \perp \ \ 1 - \boldsymbol{\zeta} \geq 0. \end{aligned} \tag{10}$$

We noted in Section 2.1 that the inner-level problems, (2), can be replaced with the first-order KKT conditions, (4–5). Furthermore, the inner-level step function LPs, (9), can be rewritten using the linear complementarity condi-

tions, (10). The overall two-level classification problem becomes

$$
\begin{aligned}
\min\ & \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|\mathcal{N}_t|}\sum_{i\in\mathcal{N}_t}\zeta_i^t \\
\text{s. t. }\ & \lambda_{\mathbf{lb}} \le \lambda \le \lambda_{\mathbf{ub}},\ \overline{\mathbf{w}}_{\mathbf{lb}} \le \overline{\mathbf{w}} \le \overline{\mathbf{w}}_{\mathbf{ub}}, \\
& \text{and for } t = 1\ldots T, \\
& \left.\begin{aligned} 0 \le \zeta_i^t \quad & \perp y_i\left(\mathbf{x}_i'\mathbf{w}^t - b_t\right) + z_i^t \ge 0 \\ 0 \le z_i^t \quad & \perp 1 - \zeta_i^t \ge 0 \end{aligned}\right\} \forall\, i \in \mathcal{N}_t, \\
& \left.\begin{aligned} 0 \le \alpha_j^t \quad & \perp y_j(\mathbf{x}_j'\mathbf{w}^t - b_t) - 1 + \xi_j^t \ge 0 \\ 0 \le \xi_j^t \quad & \perp 1 - \alpha_j^t \ge 0 \end{aligned}\right\} \forall\, j \in \overline{\mathcal{N}}_t, \\
& 0 \le \boldsymbol{\gamma}^{t,+} \perp \overline{\mathbf{w}} - \mathbf{w}^t \ge 0, \\
& 0 \le \boldsymbol{\gamma}^{t,-} \perp \overline{\mathbf{w}} + \mathbf{w}^t \ge 0, \\
& \lambda\,\mathbf{w}^t - \sum_{j\in\overline{\mathcal{N}}_t} y_j\alpha_j^t\mathbf{x}_j + \boldsymbol{\gamma}^{t,+} - \boldsymbol{\gamma}^{t,-} = 0, \\
& \sum_{j\in\overline{\mathcal{N}}_t} y_j\alpha_j^t = 0,
\end{aligned}
\tag{11}
$$

which is an instance of an MPEC. It is a nonconvex optimization problem because of the complementarity constraints. We refer to this problem as the Bilevel Misclassification Minimization (BilevelMM) problem.

3 Bilevel Classification Variations

There are many possible variations of the bilevel classification problem. To illustrate the versatility of the bilevel approach, we discuss the outer-level objective, feature selection strategies, and kernel selection.

3.1 *Outer-level objective*

The outer-level objective, (6), is not the only criterion that can be used to estimate generalization error within the cross validation scheme. An intuitively appealing alternative is to use the same misclassification measure for both the outer- and inner-level problems. Thus, we can also use the *hinge loss*, which minimizes the distance of each misclassified validation point from the classifier margin trained within each fold. The hinge loss is an upper bound on the

misclassification objective:

$$\Theta(\mathbf{W}, \mathbf{b}) = \frac{1}{T} \sum_{t=1}^{T} \frac{1}{|\mathcal{N}_t|} \sum_{i \in \mathcal{N}_t} \max\left(1 - y_i(\mathbf{x}_i' \mathbf{w}^t - b_t),\, 0\right). \tag{12}$$

The resulting MPEC is simpler because the lower level problems, (9), introduced to calculate the average number of points misclassified are not required. One might expect that this would lead to faster solutions by the FILTER solver on NEOS. But as we see later, this is not the case. When the hinge-loss is used in the outer-level, the MPEC becomes

$$\begin{aligned}
\min\ & \frac{1}{T} \sum_{t=1}^{T} \frac{1}{|\mathcal{N}_t|} \sum_{i \in \mathcal{N}_t} z_i^t \\
\text{s. t. } & \lambda_{\mathbf{lb}} \le \lambda \le \lambda_{\mathbf{ub}},\ \overline{\mathbf{w}}_{\mathbf{lb}} \le \overline{\mathbf{w}} \le \overline{\mathbf{w}}_{\mathbf{ub}}, \\
& \text{and for } t = 1 \ldots T, \\
& \left.\begin{array}{l} z_i^t \ge 1 - y_i\left(\mathbf{x}_i' \mathbf{w}^t - b_t\right) \\ z_i^t \ge 0 \end{array}\right\} \forall\, i \in \mathcal{N}_t, \\
& \left.\begin{array}{l} 0 \le \alpha_j^t \quad \perp\ y_j(\mathbf{x}_j' \mathbf{w}^t - b_t) - 1 + \xi_j^t \ge 0 \\ 0 \le \xi_j^t \quad \perp\ 1 - \alpha_j^t \ge 0 \end{array}\right\} \forall\, j \in \overline{\mathcal{N}}_t, \\
& 0 \le \boldsymbol{\gamma}^{t,+} \perp \overline{\mathbf{w}} - \mathbf{w}^t \ge 0, \\
& 0 \le \boldsymbol{\gamma}^{t,-} \perp \overline{\mathbf{w}} + \mathbf{w}^t \ge 0, \\
& \lambda\, \mathbf{w}^t - \sum_{j \in \overline{\mathcal{N}}_t} y_j \alpha_j^t \mathbf{x}_j + \boldsymbol{\gamma}^{t,+} - \boldsymbol{\gamma}^{t,-} = 0, \\
& \sum_{j \in \overline{\mathcal{N}}_t} y_j \alpha_j^t = 0.
\end{aligned} \tag{13}$$

We refer to this problem as Bilevel Hinge Loss (BilevelHL) problem.

3.2 *Enhanced feature selection*

The introduction of $\overline{\mathbf{w}}$ into the SVM represents a novel and powerful way to perform feature selection and to force $\mathbf{w}$ to be sparse. A simple way to enhance this would be to use either an L_1-norm regularization or a combination of L_1 and L_2 norms (elastic nets [28, 29]) in the inner level. These variations would only require straightforward modifications to the model. However, we will focus on yet another variation, one that attempts to incorporate prior

knowledge into feature selection.

Suppose, for n-dimensional data, it was known *a priori* that at most $n_{\mathbf{max}}$ features are sufficient. This can be incorporated into the model by introducing the constraint $\|\overline{\mathbf{w}}\|_0 \leq n_{\mathbf{max}}$ into the outer-level problem, where $\|\cdot\|_0$ is called the zero-norm or the cardinality of a vector, i.e., it counts the number of nonzero elements in its argument. This constraint forces the number of allowable features to be bounded above by some user-defined maximum and causes the features with the smallest weights to be dropped from the model. The constraint can be rewritten using the $()_\star$ function, since we have $\|\overline{\mathbf{w}}\|_0 = \mathbb{1}'\overline{\mathbf{w}}_\star$. If the conditions (10) are used to rewrite the constraint, the following inequality and complementarity constraints are added to the outer-level of (11):

$$\begin{aligned} &\sum_{m=1}^{n} \delta_m \leq n_{\mathbf{max}}, \\ &0 \leq \boldsymbol{\delta} \quad \perp -\overline{\mathbf{w}} + \mathbf{d} \geq 0, \\ &0 \leq \mathbf{d} \quad \perp \quad \mathbb{1} - \boldsymbol{\delta} \geq 0. \end{aligned} \tag{14}$$

In the constraints above, $\boldsymbol{\delta}$ counts the selected features of $\overline{\mathbf{w}}$, and $\mathbf{d}$ is the multiplier to the constraint $\mathbb{1} - \boldsymbol{\delta} \geq 0$.

3.3 *Kernel bilevel cross validation*

A fundamental limitation of (11) is the fact that it is linear and cannot handle nonlinear data sets effectively. One of the most powerful features of SVMs is their ability to deal with high-dimensional, highly nonlinear data using the *kernel trick*. We now demonstrate how a linear bilevel program can be kernelized.

An important feature of the formulation (11) is that it is capable of feature selection. However, it is clear from the first-order conditions, (5), that $\mathbf{w}^t$ depends not only on the training data, but also on the multipliers, $\boldsymbol{\gamma}^{t,\pm}$, of the box constraints. These multipliers are an impediment to expressing $\mathbf{w}^t$ solely as a linear combination of the training data, a fundamental assumption that is at the heart of all kernel methods through the representation theorem. As a consequence, temporarily setting aside the concerns of feature selection, we drop the box constraints, set $\boldsymbol{\gamma}^{t,\pm}$ to zero, and work with the classical SV classifier. The new first order conditions are

$$\lambda\, \mathbf{w}^t = \sum_{j \in \overline{\mathcal{N}}_t} y_j \alpha_j^t \mathbf{x}_j, \quad \forall\, t = 1, \ldots, T. \tag{15}$$

Now, we can eliminate $\mathbf{w}^t$ within each fold of (11) using (15). The resulting

linear inner product terms, $\mathbf{x}_i'\mathbf{x}_k$, can be replaced by a bilinear, symmetric, positive semi-definite kernel function, $\kappa(\mathbf{x}_i, \mathbf{x}_k)$. The final bilevel cross validation for kernel SV classification model is shown below with $C = 1/\lambda$:

$$
\begin{aligned}
\min\ & \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|\mathcal{N}_t|}\sum_{i\in\mathcal{N}_t}\zeta_i^t \\
\text{s. t. } & C \geq 0, \\
& \text{and for } t = 1\ldots T, \\
& \left.\begin{aligned} & 0 \leq \zeta_i^t \perp C\, y_i\left[\sum_{k\in\overline{\mathcal{N}}_t} y_k\alpha_k^t\kappa(\mathbf{x}_i, \mathbf{x}_k) - b_t\right] + z_i^t \geq 0 \\ & 0 \leq z_i^t \perp 1 - \zeta_i^t \geq 0 \end{aligned}\right\} \forall\, i \in \mathcal{N}_t, \\
& \left.\begin{aligned} & 0 \leq \alpha_j^t \perp C\, y_j\left[\sum_{k\in\overline{\mathcal{N}}_t} y_k\alpha_k^t\kappa(\mathbf{x}_j, \mathbf{x}_k) - b_t\right] - 1 + \xi_j^t \geq 0 \\ & 0 \leq \xi_j^t \perp 1 - \alpha_j^t \geq 0 \end{aligned}\right\} \forall\, j \in \overline{\mathcal{N}}_t, \\
& \sum_{j\in\overline{\mathcal{N}}_t} y_j\alpha_j^t = 0.
\end{aligned}
\tag{16}
$$

There are two new challenges raised by the kernel model. First, the kernel contains parameters that must be determined. Second, as formulated above, this model is not capable of performing feature selection. These challenges can be overcome by considering a parameterized kernel of the form $\kappa(\mathbf{x}_i, \mathbf{x}_k; \mathbf{p})$, where $\mathbf{p} \geq 0$ is the feature scaling vector whose role is similar to $\overline{\mathbf{w}}$ in the linear model; in particular, if $\mathbf{p}_i = 0$, then the i-th feature is eliminated. For example, the parameterized Gaussian kernel can be written down as below:

$$
\kappa(\mathbf{x}_i, \mathbf{x}_k; \mathbf{p}) = \exp\left(-(\mathbf{x}_i - \mathbf{x}_k)'\mathbf{diag}(\mathbf{p})\,(\mathbf{x}_i - \mathbf{x}_k)\right), \tag{17}
$$

where $\mathbf{diag}(\mathbf{p})$ is the diagonal matrix with diagonal entries given by the components of $\mathbf{p}$. Other kernels can be similarly extended and used in the model. Consequently, the components of this new vector of kernel parameters, $\mathbf{p}$, become variables in the overall bilevel kernel model. The introduction of the parameterized kernel is a very powerful extension to the linear model (11) as it is capable of picking the regularization parameters, and kernel parameters and features, leaving only the choice of kernel to the user. Additional research is needed to develop effective solvers for the bilevel kernel models. Our preliminary computational investigation found that FILTER runs through NEOS

could only successfully solve small problems. Thus, the results and discussion in this paper are limited to the linear case.

4 Inexact and Discretized Cross Validation

The bilevel formulations described in the previous section perform model selection by searching a continuous parameter space. In contrast, classical cross validation approximately solves the bilevel problem by searching a discretized version of the same parameter space. In the bilevel approach also performs inexact cross validation, by solving a relaxed version of the bilevel MPEC. Pertinent details of both these methods are described below.

4.1 *Inexact cross validation*

There exist several approaches that can deal with the complementarity constraints in MPECs such as (11). Some of these are: penalty methods, which allow for the violation of the complementarity constraints, but penalize them through a penalty term in the outer-level objective; smoothing methods, that construct smooth approximations of the complementarity constraints; and relaxation methods, that relax the complementarity constraints while retaining the convex constraints. We use the relaxation approach to solve (11).

This method of solving an MPEC simply involves replacing all instances of the "hard" complementarity constraints of the form

$$0 \leq \mathbf{c} \perp \mathbf{d} \geq 0 \quad \equiv \quad \mathbf{c} \geq 0,\, \mathbf{d} \geq 0,\, \mathbf{c}'\mathbf{d} = 0,$$

with relaxed, "soft" complementarity constraints of the form

$$0 \leq \mathbf{c} \perp_{\mathbf{tol}} \mathbf{d} \geq 0 \quad \equiv \quad \mathbf{c} \geq 0,\, \mathbf{d} \geq 0,\, \mathbf{c}'\mathbf{d} \leq \mathbf{tol},$$

where $\mathbf{tol} > 0$ is some prescribed tolerance of the complementarity conditions. This leads us to the bilevel SVC problem with *inexact cross validation*, which is the same as (11) except that all the $\perp$ conditions are replaced by $\perp_{\mathbf{tol}}$.

Even though this is still a non-convex optimization problem, it represents a novel approach in the context of machine learning. The tolerance parameter, **tol**, which is set *a priori*, determines the accuracy of the relaxation and performs inexact cross validation. That means: an appropriately chosen **tol** can enlarge the search region of the model at the expense of a tolerable decrease in model accuracy. This is similar to the well-known machine-learning concept of "early stopping" in that the quality of the out-of-sample errors—measured in the outer-level objective of the bilevel program—is not affected significantly by

small perturbations to a computed solution, in turn facilitating an early termination of cross validation. This approach also has the advantage of easing the difficulty of dealing with the disjunctive nature of the complementarity constraints. The exact same approach can also be applied to the hinge-loss MPEC, (13).

4.2 *Grid search*

Classical cross validation is performed by discretizing the parameter space into a grid and searching for the combination of parameters that minimizes the out-of-sample error, also referred to as validation error. This corresponds to the outer-level objective of the bilevel program (11). Typically, coarse logarithmic parameter grids of base 2 or 10 are used. Once a locally optimal grid point with the smallest validation error has been found, it may be refined or fine-tuned by a local search.

In the case of SV classification, the only hyper-parameter is the regularization constant, λ. However, the bilevel model (11) uses the box-constrained SVM for feature selection; Grid Search has to determine $\overline{\mathbf{w}}$ as well. It is this search in the $\overline{\mathbf{w}}$-space that causes a serious combinatorial difficulty for the grid approach. To see this, consider the case of T-fold cross validation using grid search, where λ and $\overline{\mathbf{w}}$ are each allowed to take on d discrete values. Assuming the data is n-dimensional, grid search would have to solve roughly $O(Td^{n+1})$ problems. The resulting combinatorial explosion makes grid search intractable for all but the smallest n. In this paper, to counter this difficulty, we implement the following heuristic scheme:

- To determine λ: Perform a one-dimensional grid search using the classical SVC problem (without the box constraint). The range $[\lambda_{\mathbf{lb}}, \lambda_{\mathbf{ub}}]$ is discretized into a coarse, base-10, logarithmic grid. These grid points constitute the search space for this step, which we will call *unconstrained grid search*. At each grid point, T SVC problems are solved on the training sets and the error is measure on the validation sets. The grid point with the smallest average validation error, $\overline{\lambda}$, is "optimal".
- To determine $\overline{\mathbf{w}}$: Perform a n-dimensional grid search to determine the relevant features of $\overline{\mathbf{w}}$ using the box-constrained SVC problem (BoxSVC) and $\overline{\lambda}$ obtained from the previous step. Only two distinct choices for each feature are considered: 0, to test feature redundancy, and some large value that would not affect the choice of an appropriate feature weight. In this setting, 3-fold cross validation would involve solving about $O(3 * 2^n)$ BoxSVC problems. We call this step *constrained grid search*.
- The number of problems in constrained grid search is already impractical necessitating a further restriction of the relevant features to a maximum of

Table 1. Descriptions of data sets used.

Data set	ℓ_{train}	ℓ_{test}	n	T
Pima Indians Diabetes Database	240	528	8	3
Wisconsin Breast Cancer Database	240	442	9	3
Cleveland Heart Disease Database	216	81	14	3
Johns Hopkins University Ionosphere Database	240	111	33	3
Star/Galaxy Bright Database	600	1862	14	3
Star/Galaxy Dim Database	900	3292	14	3

$n = 10$. If a data set has more features, they are ranked using *recursive feature elimination* [30], and the 10 best features are chosen.

5 Numerical Tests

We compare unconstrained and constrained grid search approaches to the bilevel approaches (11) and (13) relaxed through inexact cross validation. The bilevel programs were implemented in AMPL and solved using FILTER [31–33], which is a general-purpose nonlinear programming solver available on the NEOS server (www-neos.mcs.anl.gov). Unconstrained and constrained grid were solved using MOSEK's quadratic program solver accessed through a MATLAB interface.

5.1 *Experimental design*

We used 6 real-world classification data sets, four of which are available via anonymous ftp from the UCI Repository for Machine Learning and two from the Star/Galaxy database at the University of Minnesota. The data sets were all standardized to zero norm and unit standard deviation. Twenty instances of each data set were randomly generated and each instance was split into a training set with ℓ_{train} points, which is used for cross validation and a hold-out test set, with ℓ_{test} points. The data descriptions are shown in Table 5.1. The hyper-parameters in the bilevel program were restricted as follows: $\lambda \in [10^{-4}, 10^{4}]$ and $\overline{\mathbf{w}} \in [0, 1.5]$. Grid search used the exact same bounds but was further restricted to $\lambda \in \{10^{-4}, 10^{-3}, \ldots, 10^{3}, 10^{4}\}$ and $\overline{\mathbf{w}} \in \{0, 1.5\}$. The complementarity tolerance was set to be $\mathbf{tol} = 10^{-6}$ in all runs except in the BilevelHL problem on the `dim` data set, where the value of $\mathbf{tol} = 10^{-4}$ was used. These settings were used to perform 3-fold cross validation on each instance.

Using the cross-validated hyper-parameters $\widehat{\lambda}$ and $\widehat{\overline{\mathbf{w}}}$ obtained from the bilevel and the grid search approaches, we implement a post-processing procedure to calculate the generalization error on the hold-out data for each instance. Specifically, a constrained SVC problem is solved on *all* the training data using $\frac{3}{2}\widehat{\lambda}$ and $\widehat{\overline{\mathbf{w}}}$ giving the final classifier $(\widehat{\mathbf{w}}, \widehat{b})$ which is used to compute

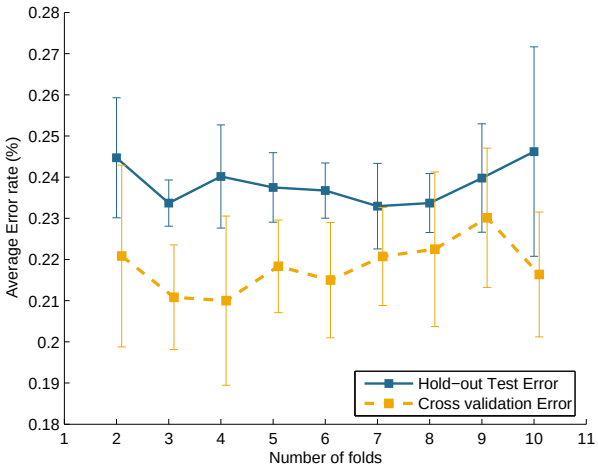

Figure 1. Effect of increasing the number of folds on learning rate of classifiers for `pima`.

the test (hold-out) error rate:

$$ERROR_{test} = \frac{1}{\ell_{test}} \sum_{(\mathbf{x},y)\ \text{test}} \frac{1}{2} |\ \mathbf{sign}(\widehat{\mathbf{w}}'\mathbf{x} - \widehat{b}) - y\ |.$$

Recall that the bilevel model uses 3-fold cross-validation and that each training fold consists of two-thirds of the total training data. Consequently, the final regularization parameter, $\widehat{\lambda}$, is rescaled by a factor of $\frac{3}{2}$ because the final model, which is constructed in the post-processing phase, uses all of the training data. For general T-fold cross validation, as mentioned before, this factor will be $\frac{T}{T-1}$, $T > 1$, assuming that each fold contains the same fraction of data.

In addition, we also compute the cardinality of the final $\overline{\mathbf{w}}$ returned by the different approaches to determine the effectiveness of feature selection. For the bilevel approaches, the features in $\overline{\mathbf{w}}$ with weights less than $\sqrt{\mathbf{tol}}$ were considered irrelevant and set to zero, *after* which the test error was computed. Various criteria are used to compare the bilevel approach to the grid search approach: cross-validation error, test error, feature selection and execution time. The results, averaged over 20 instances for each data set, are presented in Table 5.2. Results which are significantly different (using a paired t-test at 10% confidence) with respect to unconstrained grid are shown in bold. The computational results in Table 5.2 all used $T = 3$ cross validation folds.

To study the effect of increasing the number of folds on cross validation error and test error, we report, in Figure 1, the results averaged over 5 instances of the `pima` data set. The results clearly demonstrate that larger number of folds

Table 2. Computational Results comparing Grid Search and Bilevel Approaches.

Data set	Method	CV Error	Test Error	$\|\overline{\mathbf{w}}\|_0$	Time (sec.)
`pima`	Unconstrained Grid	23.10 ± 2.12	23.75 ± 0.94	8.0	12.3± 1.1
	Constrained Grid	$\mathbf{21.04 \pm 1.63}$	24.13 ± 1.13	4.5	434.9± 35.1
	Bilevel (Misclass Min)	$\mathbf{21.87 \pm 2.25}$	23.90 ± 0.95	6.4	51.8± 23.0
	Bilevel (HingeLoss Min)	$\mathbf{44.04 \pm 3.19}$	23.80 ± 1.14	5.4	156.5± 57.5
`cancer`	Unconstrained Grid	3.54 ± 1.14	3.61 ± 0.64	9.0	11.7± 0.4
	Constrained Grid	$\mathbf{2.73 \pm 0.88}$	$\mathbf{4.42 \pm 0.85}$	5.8	815.5± 29.7
	Bilevel (Misclass Min)	$\mathbf{3.13 \pm 1.05}$	3.59 ± 0.79	8.2	19.9± 8.3
	Bilevel (HingeLoss Min)	$\mathbf{6.13 \pm 2.22}$	3.76 ± 0.74	6.8	58.3± 33.6
`heart`	Unconstrained Grid	15.93 ± 2.02	16.05 ± 3.65	13.0	10.6± 0.8
	Constrained Grid	$\mathbf{13.94 \pm 1.69}$	16.85 ± 4.15	7.1	1388.7± 37.6
	Bilevel (Misclass Min)	$\mathbf{14.49 \pm 1.47}$	16.73 ± 3.89	11.2	64.0± 20.5
	Bilevel (HingeLoss Min)	$\mathbf{28.89 \pm 3.20}$	16.30 ± 3.29	8.8	217.1± 82.5
`ionosphere`	Unconstrained Grid	22.27 ± 2.45	23.06 ± 2.45	33.0	7.5± 0.6
	Constrained Grid	$\mathbf{19.25 \pm 2.07}$	22.34 ± 2.02	6.9	751.1± 3.0
	Bilevel (Misclass Min)	$\mathbf{19.16 \pm 2.44}$	23.65 ± 2.99	20.2	423.0±159.5
	Bilevel (HingeLoss Min)	$\mathbf{33.79 \pm 2.79}$	22.79 ± 2.03	14.2	1248.8±618.5
`bright`	Unconstrained Grid	0.78 ± 0.34	0.74 ± 0.13	14.0	22.7± 0.2
	Constrained Grid	$\mathbf{0.51 \pm 0.24}$	$\mathbf{0.97 \pm 0.33}$	6.7	3163.7± 11.5
	Bilevel (Misclass Min)	$\mathbf{0.62 \pm 0.31}$	0.79 ± 0.14	11.2	110.9± 61.2
	Bilevel (HingeLoss Min)	$\mathbf{1.12 \pm 0.58}$	0.75 ± 0.14	8.9	564.2±335.7
`dim`	Unconstrained Grid	4.71 ± 0.55	4.96 ± 0.29	14.0	55.0± 5.1
	Constrained Grid	$\mathbf{4.36 \pm 0.51}$	$\mathbf{5.21 \pm 0.37}$	7.2	7643.5± 74.5
	Bilevel (Misclass Min)	4.77 ± 0.64	$\mathbf{5.51 \pm 0.33}$	7.7	641.5±344.1
	Bilevel (HingeLoss Min)	$\mathbf{9.54 \pm 1.00}$	$\mathbf{5.28 \pm 0.36}$	5.7	1465.2±552.9

can be successfully solved, but computation time does grow with the number of folds. The range of generalization values observed for different numbers of folds is not large, so $T = 3$ represents a reasonable choice. The best choice of the number of folds for a particular data set remains an open question.

5.2 *Discussion*

We first examine the performance of the bilevel misclassification minimization (BilevelMM) programming approach with respect to the grid search methods. The first conclusion that can be drawn, from computational efficiency perspective, is that BilevelMM vastly outperforms the constrained grid search approach; the execution times for the former are several orders of magnitude smaller than the latter. It should be noted that the reported computation times for FILTER include transmission times as well as solve times, and that the reported computation times for grid search are enhanced by the use of smart restarting heuristics. However, despite the latter, it is clear that constrained grid search quickly becomes impractical as the problem size grows. This effect is clearly noticeable in the computation times for the Star/Galaxy data sets, where the execution time is affected, not only by the number of fea-

tures, but also by the data set sizes, which contain hundreds of training points. BilevelMM, on the other hand, is capable of cross-validating the `dim` data set, with 900 training points, in around 10-11 minutes on average. This suggests that the scalability of the bilevel approach could be improved significantly by exploiting the structure and sparsity inherent in SVMs. Research is currently underway in this direction and findings will be reported elsewhere.

With regard to generalization error, two interesting points emerge. First, the BilevelMM approach consistently produces results that are comparable to, if not slightly better than the grid search approaches, in spite of the fact that the cross-validation error is typically higher. This can be attributed to the fact that general-purpose NLP solvers tend to converge to acceptable solutions, with no guarantee of global optimality. Second, the only exception is the `dim` data set where the slight degradation in generalization performance can be imputed to numerical difficulties experienced by the NLP solvers because of large dimensionality and large data set size. Again, a specialized algorithm that could guarantee global optimality could produce better generalization performance.

With regard to feature selection, it is clear that unconstrained grid performs none at all, while, interestingly, constrained grid search uses less features than BilevelMM, albeit at the expense of excessive computational times and poorer generalization. This can be attributed to the fact that constrained grid is greedy, i.e., it analyzes *every* combination of features to find an optimal set and performs feature selection aggressively on data sets with more than 10 features as it drops the remaining features using RFE. BilevelMM has no such heuristic or mechanism to drive the number of selected features down. Despite this, it is clear that it does succeed in performing a better trade-off between feature selection and generalization. See Section 3.2 for ideas that might improve feature selection in the bilevel setting.

Next, we discuss the performance of the bilevel hinge-loss (BilevelHL) approach and compare it to BilevelMM. The most striking difference is in the computation times of the two approaches, with BilevelHL, quite unexpectedly, taking two to three times longer. We theorize that this is because BilevelMM has many more stationary points (for an intuitive explanation of this curious property that is endemic to misclassification minimization problems, see [26]) than BilevelHL and consequently, a general-purpose NLP solver tends to converge to stationarity faster. However, BilevelHL is still considerably faster than grid search; again, the only exception being the `ionosphere` data set. It should be noted that constrained grid search used only 10 features—after recursive feature elimination was used to drop 23 of the 33 features—while BilevelHL solved the full problem using all the features. If constrained grid were to use all 33 features, it would have to solve around $O(10^{11})$ BoxSVC problems.

In terms of generalization error, BilevelHL performs as well or better than

BilevelMM and never significantly worse than unconstrained grid (except for the `dim` data set), despite the fact that the CV errors of BilevelHL are uniformly higher. Recall that a complementarity tolerance of 10^{-4} was used for the `dim` data set. This was to relax the problem further for the numerical stability of the NLP solver. This relaxation, however, leads to a slight degradation in the quality of the solution.

Finally, BilevelHL tends to pick fewer features than BilevelMM, but still more than constrained grid. This comparison between BilevelMM and BilevelHL indicates that the best choice of outer-level objective is still an open question in need of further research.

6 Conclusions

We demonstrated how T-fold cross-validation can be cast as a continuous bilevel program: inner-level problems are introduced for each of the T-folds to compute classifiers on the training sets and to calculate the misclassification errors on the training sets within each fold. Furthermore, we introduced the box-constrained SVM which has a hyper-parameter for each feature to perform feature selection. The resulting bilevel program is converted to an MPEC, which is in turn converted to a nonlinear programming problem through inexact cross validation. The advantage of the bilevel approach is that many hyper-parameters can be optimized simultaneously, unlike prior grid search approaches that are practically limited to one or two parameters. Initial computational results using FILTER through NEOS were very promising. High quality solutions were found using few features using much less computation time than grid search approaches over the same hyper-parameters.

This work represents a first proof of concept. We showed that cross-validation through minimization of different objectives such as averaged misclassification error and hinge loss could be solved efficiently with large numbers of hyper-parameters. The resulting classifiers demonstrated good generalization ability and were dependent on only a few features. The success of these two different bilevel approaches suggests that other changes in the objective and regularization can lead to further enhancement of performance for classification problems. Furthermore, the versatility of the bilevel approach suggests that further variations could be developed to tackle other challenges in machine learning such as missing data, semi-supervised learning, kernel learning and multi-task learning. Future theoretical and computational work is needed to investigate this flexibility–that the bilevel approach has the ability to optimize large number of hyper-parameters for many types of outer-level objectives.

A major outstanding research question is the development of efficient optimization algorithms for the bilevel program. Our current work is limited to

the use of an off-the-shelf NLP solver. A recent linear time SVM algorithm can solve traditional SVM classification problems with millions of data points [34] using advanced decomposition techniques that exploit the underlying structure of the problem. Many other efficient and scalable methods for SVM abound and these methods should be compared with and incorporated into the bilevel approach [35, 36]. Solution path algorithms traverse the feasible regions for very limited bilevel problems. The hope is that by developing related special purpose solvers the scalability of the bilevel program can be achieved as well.

Acknowledgement

This work was supported in part by the Office of Naval Research under grant no. N00014-06-1-0014.

References

[1] Cortes, C. and Vapnik, V., 1995, Support-vector networks. *Machine Learning,* **20**, 273–297.
[2] Vapnik, V., 2000, *The Nature of Statistical Learning Theory* (2nd edn) (New York, NY: Springer-Verlag).
[3] Chapelle, O, Vapnik, V., Bousquet, O. and Mukherjee, S., 2002, Choosing multiple parameters for support vector machines. *Machine Learning,* **46**, 131–159.
[4] Duan, K., Keerthi, S. and Poo, A., 2003, Evaluation of simple performance measures for tuning SVM hyperparameters. *Neurocomputing,* **51**, 41–59.
[5] Hastie, T., Rossett, S., Tibshirani, R. and Zhu, J., 2004, The entire regularization path for the support vector machine. *Journal of Machine Learning Research,* **5**, 1391-1415.
[6] Bennett, K. P., Ji, X., Hu, J., Kunapuli, G. and Pang, J.-S., 2006, Model selection via bilevel programming. *Proceedings of the IEEE International Joint Conference on Neural Networks,* Vancouver, B.C., Canada, July 16–21, 1922–1929.
[7] Golub, G., Heath, M. and Wahba, G., 1979, Generalised cross validation as a method for choosing a good ridge parameter. *Technometrics,* **21**, 215–223.
[8] Bi, J., Bennett, K. P., Embrechts, M., Breneman, C. and Song, M., 2003, Dimensionality reduction via sparse support vector machines. *Journal on Machine Learning Research,* **3**, 1229–1243.
[9] Shawe-Taylor, J. and Cristianini, N., 2004, *Kernel Methods for Pattern Analysis* (Cambridge, UK: Cambridge University Press).
[10] Zhu, J., Rosset, S., Hastie, T. and Tibshirani, R., 2003, 1-norm support vector machines. *Advances in Neural Information Processing Systems,* **16**.
[11] Wang, G., Yeung, D. and Lochovsky, F., 2006, Two-dimensional solution path for support vector regression. *Proceedings of the 23th International Conference on Machine Learning,* **148**, 993–1000.
[12] Luo, Z. Q., Pang, J.-S. and Ralph, D., 1996, *Mathematical Programs With Equilibrium Constraints* (Cambridge, UK: Cambridge University Press).
[13] Anitescu, M., 2005, On solving mathematical programs with complementarity constraints as nonlinear programs. *SIAM Journal on Optimization,* **15**, 1203–1236.
[14] Anitescu, M., Tseng, P. and Wright, S. J., 2005, Elastic-mode algorithms for mathematical programs with equilibrium constraints: Global convergence and stationarity properties. *Preprint ANL/MCS-P1242-0405,* Mathematics and Computer Science Division, Argonne National Laboratory.
[15] Dempe, S., 2002, *Foundations of Bilevel Programming.* (Dordrecht, The Netherlands: Kluwer Academic Publishers).
[16] Dempe, S., 2003, Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints. *Optimization,* **52**, 333–359.

[17] Fletcher, R. and Leyffer, S., 2004, Solving mathematical programs with complementarity constraints as nonlinear programs. *Optimization Methods and Software,* **19**, 15-40.
[18] Fletcher, R., Leyffer, S., Ralph, D. and Scholtes, S., 2006, Local convergence of SQP methods for mathematical programs with equilibrium constraints. *SIAM Journal on Optimization,* **1**, 259–286.
[19] Facchinei, F. and Pang, J.-S., 2003, *Finite-Dimensional Variational Inequalities and Complementarity Problems,* (New York, NY: Springer-Verlag).
[20] Fukushima, M. and Pang, J.-S., 1999, Convergence of a smoothing continuation method for mathematical programs with complementarity constraints. In: Michel Théra and Rainer Tichatschke (Eds) *Ill-posed variational problems and regularization techniques: Lecture Notes in Economics and Mathematical Systems,* **477** (Berlin, Germany: Springer), pp. 99–110.
[21] Fukushima, M. and Tseng, P., 2002, An implementable active-set algorithm for computing a B-stationary point of the mathematical program with linear complementarity constraints. *SIAM Journal on Optimization,* **12**, 724–739.
[22] Jiang, H. and Ralph, D., 2000, Smooth SQP methods for mathematical programs with nonlinear complementarity constraints. *SIAM Journal on Optimization,* **10**, 779–808.
[23] Scheel, H. and Scholtes, S., 2000, Mathematical programs with complementarity constraints: Stationarity, optimality, and sensitivity. *Mathematics of Operations Research,* **25**, 1–22.
[24] Scholtes, S. and Stöhr, M., 1999, Exact penalization of mathematical programs with equilibrium constraints. *SIAM Journal on Control and Optimization,* **37**, 617–652.
[25] Scholtes, S., 2001, Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. *SIAM Journal on Optimization,* **11**, 918–936.
[26] Mangasarian, O., L., 1994, Misclassification minimization. *Journal of Global Optimization,* **5**, 309–323.
[27] Kohavi, R. and John, G., 1997, Wrappers for feature subset selection. *Artificial Intelligence,* **97**, 273-324.
[28] Tibshirani, R., 1996, Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B,* **58**, 267-288.
[29] Zou, H. and Hastie, T., 2005, Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B,* **67(2)**, 301–320.
[30] Guyon, I., Weston, J., Barnhill, S. and Vapnik, V., 2002, Gene selection for cancer classification using support vector machines. *Machine Learning,* **46**, 389–422.
[31] Fletcher, F. and Leyffer, S., 2002, Nonlinear programming without a penalty function. *Mathematical Programming,* **91**, 239-270.
[32] Fletcher, R. and Leyffer, S., 1999, User manual for FilterSQP. Available online at http://www-unix.mcs.anl.gov/leyffer/papers/SQP_manual.pdf (accessed 10 October 2006).
[33] Fletcher, R., Leyffer, S. and Toint, Ph. L., 2002, On the global convergence of a filter-SQP algorithm. *SIAM J. Optimization,* **13**, 44-59.
[34] Joachims, T., 2006, Training SVMs in Linear Time. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* 217–226.
[35] Ferris, M. and Munson, T., 2004, Semismooth support vector machines. *Mathematical Programming,* **101**, 185–204.
[36] Scheinberg, K., 2006, An efficient implementation of an active set method for SVMs. *Journal of Machine Learning Research,* **7**, 2237–2257.