Sparse Penalized Forward Selection for Support Vector Classification

Subhashis Ghosal, Bradley Turnnbull,
Department of Statistics, North Carolina State University
Hao Helen Zhang
Department of Mathematics and Statistics, University of Arizona
and Wook Yeon Hwang
LG Electronics, Seoul, Republic of Korea

Abstract

We propose a new binary classification and variable selection technique especially designed for high dimensional predictors. Among many predictors, typically, only a small fraction of them have significant impact on prediction. In such a situation, more interpretable models with better prediction accuracy can be obtained by variable selection along with classification. By adding an ℓ_1-type penalty to the loss function, common classification methods such as logistic regression or support vector machines (SVM) can perform variable selection. Existing penalized SVM methods all attempt to jointly solve all the parameters involved in the penalization problem altogether. When data dimension is very high, the joint optimization problem is very complex and involves a lot of memory allocation. In this article, we propose a new penalized forward search technique which can reduce high-dimensional optimization problems to one dimensional optimization by iterating the selection steps. The new algorithm can be regarded as a forward selection version of the penalized SVM and its variants. The advantage of optimizing in one dimension is that the location of the optimum solution can be obtained with intelligent search by exploiting convexity and a piecewise linear or quadratic structure of the criterion function. In each step, the predictor which is most able to predict the outcome is chosen in the model. The search is then repeatedly used in an iterative fashion until convergence occurs. Comparison of our new classification rule with L_1-SVM and other common methods show very promising performance, in that the proposed method leads to much leaner models without compromising misclassification rates, particularly for high dimensional predictors.

KEYWORDS: High Dimension; Sparsity; SVM; Penalization; Variable Selection.

1 Introduction

Classification is a fundamental supervised learning problem, where objects are to be classified into one of two (or more) categories based on the value of an auxiliary variable and training data. In other words, based on a sampled dataset, we wish to obtain a rule $f : \mathbb{R}^d \to \{-1, 1\}$ that will classify a future object with observed value $\boldsymbol{x}$ to the class $f(\boldsymbol{x})$. Within the available data, typically one part is used to estimate the rule f, and is called the training data. The other part of the data is used to test the accuracy of the rule, and is known as the test data. The proportion of misclassified objects in the test data is known as test error, and is instrumental in comparing different classification rules. As the rule f is unknown to us, typically a reasonable functional form of f is proposed, and the parameters in f are estimated using the training data, denoted by $(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n) \in \mathbb{R}^d \times \{-1, 1\}$. A natural strategy is to consider a loss function which penalizes wrong classification.

Classical techniques for classification like logistic regression or linear discriminant analysis use a model for the joint distribution of $(\boldsymbol{X}, Y)$, or the conditional distribution of $(Y|\boldsymbol{X})$. These method estimate the probabilities $\mathrm{P}(Y = \pm 1|\boldsymbol{X} = \boldsymbol{x})$ and derive a classification rule based on how these probabilities compare with $\frac{1}{2}$. The method of support vector machines (SVM), on the other hand, model the classification function $f(\boldsymbol{x})$ as $\mathrm{sign}(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{x})$, and then estimate β_0 and $\boldsymbol{\beta}$ by maximizing the "margin", which is essentially the distance from the boundary to the closest points, provided that the two classes completely separate out by a hyperplane (the linearly separable case). In general, slack variables $\varepsilon_1, \ldots, \varepsilon_n$ are introduced to compensate for observations falling in the wrong side of the boundary, and the SVM method minimizes

$$\frac{1}{2}\|\boldsymbol{\beta}\|^2 + \gamma \sum_{i=1}^{n} \varepsilon_i \quad \text{subject to } Y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{X}_i) \geq 1 - \varepsilon_i,\ i = 1, \ldots, n. \tag{1}$$

By convex duality, the optimization problem is equivalent to the minimization of

$$\sum_{i=1}^{n} \{1 - Y_i(\beta_0 + \boldsymbol{\beta}^T\boldsymbol{X}_i)\}_+ + \lambda\|\boldsymbol{\beta}\|^2, \tag{2}$$

where $\lambda > 0$ is a tuning parameter and $\|\boldsymbol{\beta}\|^2 = \sum_{j=1}^{d} \beta_j^2$, $\boldsymbol{\beta}^T = (\beta_1, \ldots, \beta_d)$. Here and throughout the paper, a_+ will denote $\max(a, 0)$.

In many modern day applications in bioinformatics, genomics, DNA microarray analysis, functional magnetic resonance imaging (fMRI), metabolism data and other fields, the dimension d of

the covariate space is often very high. In such situations, in addition to the classification issue, variables need to be selected as well. Variable selection gives a much simpler model for prediction and leads to more stable estimates of the parameters. Moreover, a simpler model is much easier to interpret and usually has better predicting power, by avoiding overfitting the training data. Variable selection is typically achieved by using a penalty function whose minimization has some sparse representation. A classic example of such a procedure is the least absolute deviation shrinkage and selection operator (LASSO) for linear regression (Tibshirani 1996), which minimizes

$$\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_1, \tag{3}$$

where $\|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{d} |\beta_j|$. The neighborhoods induced by the ℓ_1-norm $\|\cdot\|_1$ are diamond shaped, and hence the minimizer may have several zero components, effectively eliminating the corresponding variables.

For the classification problem, the LASSO is not among the most appropriate variable selection techniques, such that it is designed for a continuous response variable Y. However, the ℓ_1-penalization can be imposed on the penalty function in the SVM, leading to a sparse SVM (Zhu et al. 2004) defined as the minimizer of

$$\sum_{i=1}^{n} \{1 - Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{X}_i)\}_+ + \lambda\|\boldsymbol{\beta}\|_1. \tag{4}$$

Alternatively, Zhang et al. (2006) proposed the SCAD SVM which minimizes the following penalized hinge loss

$$\sum_{i=1}^{n} \{1 - Y_i(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{X}_i)\}_+ + \sum_{j=1}^{d} p_\lambda|\boldsymbol{\beta}_j|, \tag{5}$$

where p_λ is the non-concave SCAD penalty (Fan and Li 2001). Liu and Wu (2007) proposed using a combination of ℓ_0- and ℓ_1-penalty, and Zou (2007) considered the adaptive ℓ_1-penalty.

Recently, Hwang, Zhang and Ghosal (2009) proposed a penalized forward selection technique for variable selection in linear regression. The procedure, called the forward iterative regression and shrinkage technique (FIRST), exploits the analytical expressions for one-dimensional optimizers of $\sum_{i=1}^{n}(Y_i - bX_{ij})^2 + \lambda|b|$ given by $\hat{\beta}_j^L = (\hat{\beta}_j - \lambda/2)_+\text{sign}(\hat{\beta}_j)$, where $\hat{\beta}_j$ stands for the ordinary least squares estimator for regression of Y on the jth predictors X_j. The jth predictor, where j minimizes $\sum_{i=1}^{n}(Y_i - \hat{\beta}_j^L X_{ij})^2$, is then included in the model and the process is repeated by replacing Y_i by $Y_i - \hat{\beta}_j^L X_{ij}$. The procedure is stopped when the reduction in prediction error falls below a pre-

determined level. The method typically leads to smaller prediction error and substantially sparser models than common methods like the LASSO. In addition, if the sample size n is large, FIRST takes only a fraction of the time LASSO takes to compute using its popular LARS algorithm (Efron et al. 2004). Moreover, the componentwise approach to the minimization of the objective function extends to similar optimization problems such as the nonnegative garrote (Breiman 1995), elastic net (Zou and Hastie 2005), adaptive LASSO (Zou 2006; Wang et al. 2007), and glmnet (Friedman et al. 2007; Friedman et al. 2010).

Existing penalized SVM methods, including ℓ_1-norm SVM and SCAD-SVM, all attempt to directly handle a large-scaled optimization problem by solving for all the parameters involved in the model simultaneously. When the data dimension is very high, this joint optimization problem may require a huge memory allocation. In this article, we explore the idea of selecting one variable at a time to minimize a one-dimensional analog of (4) or its variants. For each predictor j, we determine the best coefficient of X_j in describing the supporting line for optimal classification of Y using a one-predictor SVM based on X_j. Further, an absolute value penalty filters out variables that have insignificant effect in classifying observations. We then select the best predictor which minimizes the prediction error given by the loss function. Once a predictor is used in the model, its linear effect is subtracted from the observation before proceeding to the next step. The process is iterated until improvement ceases to be significant. As it will turn out, unlike the linear regression case, even the one-dimensional minimization problem does not have a closed form solution. However, we shall exploit certain structural properties in the loss function and its derivative with respect to the optimizer to narrow down the search for the optimizer using a recursive algorithm, which terminates optimally. The proposed optimization procedure is more efficient in locating the minimizer than a general optimization procedure like linear or quadratic programming in the resulting one-dimensional optimization problem.

This article is organized as follows. In Section 2, we present the main idea of our method and describe its computational algorithm. A variation of the new method is considered and fully developed in Section 3. Simulation results are presented and discussed in Section 4. In Section 5, we illustration the performance of the new methods with a real microarray gene expression dataset. Some final remarks are given at the end.

2 Description of the proposed method

In this section, we give a precise technical description of the proposed method which selects predictors in the classification rule one at a time as in a forward selection process by minimizing a one-dimensional analog of (4).

Without loss of generality, we assume that all the predictors are centered and scaled to have norm one. Next, the most appropriate value of the intercept is chosen by minimizing (4) with $\boldsymbol{\beta}$ set to the zero vector, i.e., we minimize

$$\sum_{i=1}^{n}(1-Y_i b)_+ = (1-b)_+ \#\{i : Y_i = 1\} + (1+b)_+ \#\{i : Y_i = -1\}. \tag{6}$$

The minimizer of (6) is given by

$$b_0 = \text{sign}(N_+ - N_-) \tag{7}$$

assuming that $N_+ := \#\{i : Y_i = 1\} > 0$ and $N_- := \#\{i : Y_i = -1\} > 0$. The proof of (7) is given in the appendix. Once the intercept is selected, its effect is subtracted from the observation, leading to $c_i = Y_i = b_0$.

To select the predictors one by one, we evaluate the potential effect of each predictor on explaining the observation Y. For the jth predictor, $j = 1, \ldots, d$, we consider the minimization of

$$F(b) = \sum_{i=1}^{n}(c_i - Y_i b X_{ij})_+ + \lambda|b| = \sum_{i:Y_i=1}(c_i - bX_{ij})_+ + \sum_{i:Y_i=-1}(c_i + bX_{ij})_+ + \lambda|b|. \tag{8}$$

The resulting solution is given by the unique b with

$$F'(b-) \leq 0 \leq F'(b+), \tag{9}$$

where

$$\begin{aligned} F'(b+) \;=\; & -\sum_{i:Y_i=1, X_{ij}>0, b<c_i X_{ij}^{-1}} X_{ij} - \sum_{i:Y_i=1, X_{ij}<0, b\geq c_i X_{ij}^{-1}} X_{ij} \\ & + \sum_{i:Y_i=-1, X_{ij}>0, b\geq -c_i X_{ij}^{-1}} X_{ij} + \sum_{i:Y_i=-1, X_{ij}<0, b<-c_i X_{ij}^{-1}} X_{ij} + \lambda\,\text{sign}(b+) \end{aligned} \tag{10}$$

and

$$F'(b-) = -\sum_{i:Y_i=1,X_{ij}>0,b\le c_iX_{ij}^{-1}} X_{ij} - \sum_{i:Y_i=1,X_{ij}<0,b>c_iX_{ij}^{-1}} X_{ij} + \sum_{i:Y_i=-1,X_{ij}>0,b>-c_iX_{ij}^{-1}} X_{ij} + \sum_{i:Y_i=-1,X_{ij}<0,b\le -c_iX_{ij}^{-1}} X_{ij} + \lambda\,\mathrm{sign}(b-). \tag{11}$$

The proofs of (9)–(11) are given in the appendix. Let us denote the solution by $\hat{\beta}_j$.

Among different possible choices of j, we choose the one which reduces the objective function $\sum_{i=1}^n (c_i - \hat{\beta}_j X_j)_+$ the most. Once the best predictor j is selected, we replace c_i by $c_i - \hat{\beta}_j X_j$, and redo the previous step. The procedure is continued until the reduction of the objective function ceases to be significant. If we disallow the participation of a predictor j for which $\hat{\beta}_j = 0$ in the subsequent steps, then this aggressive version can save huge computing time, especially in very high dimensional models, by removing insignificant predictors permanently from the pool.

The computation of (10) and (11) from scratch involves $O(n^2)$ calculations — $O(n)$ calculations for each of the (at most) n "knot points" $\{c_iY_iX_{ij}^{-1} : i = 1,\ldots,n\}$, and hence locating the solution of (9) will be slow. The computational burden can be reduced to $O(n)$ easily by observing that (10) and (11) corresponding to adjacent knots can be related by a recurrence relation described below:

$$F'(0+) = -\sum_{i:Y_i=1,X_{ij}>0,c_iX_{ij}^{-1}>0} X_{ij} - \sum_{i:Y_i=1,X_{ij}<0,c_iX_{ij}^{-1}\le 0} X_{ij} + \sum_{i:Y_i=-1,X_{ij}>0,c_iX_{ij}^{-1}\ge 0} X_{ij} + \sum_{i:Y_i=-1,X_{ij}<0,c_iX_{ij}^{-1}<0} X_{ij} + \lambda \tag{12}$$

and

$$F'(0-) = -\sum_{i:Y_i=1,X_{ij}>0,c_iX_{ij}^{-1}\ge 0} X_{ij} - \sum_{i:Y_i=1,X_{ij}<0,c_iX_{ij}^{-1}<0} X_{ij} + \sum_{i:Y_i=-1,X_{ij}>0,c_iX_{ij}^{-1}>0} X_{ij} + \sum_{i:Y_i=-1,X_{ij}<0,c_iX_{ij}^{-1}\le 0} X_{ij} - \lambda. \tag{13}$$

Order all knots $\{c_iY_iX_{ij}^{-1} : i = 1,\ldots,n\}$. If ξ is a negative knot or zero, and $T = F'(\xi-)$, then

for ξ^* the knot on its left closest to ξ, $T^* = F'(\xi^*-)$ can be calculated by

$$T^* = T - \sum_{i:c_i Y_i X_{ij}^{-1} = \xi^*} |X_{ij}|. \tag{14}$$

On the other hand, if ξ is a positive knot or zero, and $T = F'(\xi+)$, then for ξ^* the knot on its right closest to ξ, $T^* = F'(\xi^*+)$ can be calculated by

$$T^* = T + \sum_{i:c_i Y_i X_{ij}^{-1} = \xi^*} |X_{ij}|. \tag{15}$$

The proofs of (14) and (15) are given in the appendix.

We note that the calculation of the left and right derivatives at all points plus zero is not essential. A substantial saving in computation is possible by computing only those values which are actually necessary to locate the minima. Since F is a convex and piecewise linear function, F' exists at all points except at the knot points and zero, and F' is increasing, piecewise constant, starting with negative values from the left, changing sign only once. Thus if $F'(0-) \leq 0 \leq F'(0+)$, $b = 0$ is the minima, and the search will stop. This will happen with all variables which do not have a significant role in predicting Y. If $F'(0-) > 0$, the search for the minima should be on the negative side only, so (14) should be repeated until T^* becomes negative. Thus the first instance when T^* becomes negative, the minima will be at the corresponding ξ^*. On the other hand, if $F'(0+) < 0$, the search for the minima should be on the positive side only, so (15) should be repeated until T^* becomes positive. Thus the first instance when T^* becomes positive, the minima will be at the corresponding ξ^*.

The whole procedure can be described by the following algorithm. Here we fix a precision parameter $\epsilon > 0$. For any given value of the tuning parameter $\lambda > 0$, do the following steps:

Algorithm 1

1. Standardization step: Replace X_{ij} by $(X_{ij} - \bar{X}_j)/\{\sum_{i=1}^n (X_{ij} - \bar{X}_j)^2\}^{1/2}$ for $i = 1, \ldots, n$, and $j = 1, \ldots, d$, where $\bar{X}_j = n^{-1} \sum_{i=1}^n X_{ij}$. Thus from now on, we shall assume that $\bar{X}_j = 0$ and $\sum_{i=1}^n X_{ij}^2 = 1$.

2. Intercept estimation step: Compute $N_+ = \#\{i : Y_i = 1\}$ and $N_- = \#\{i : Y_i = -1\}$, and set $\hat{\beta}_0 = \text{sign}(N_+ - N_-)$. Compute $S_0 = \sum_{i=1}^n (1 - Y_i \hat{\beta}_0)_+^2$ and set $m = 1$ and $c_{i,m} = 1 - Y_i \hat{\beta}_0$.

3. Optimization step: For each $j = 1, \ldots, d$, do the following:

   (i) Compute $F'(0+)$ and $F'(0-)$ respectively by (12) and (13).

   (ii) If $F'(0-) \leq 0 \leq F'(0+)$, set $\hat{\beta}_{j,m} = 0$ and stop. [For the aggressive version, remove j permanently from the pool of possible predictors.]

   (iii) Else if $F'(0-) > 0$, order all negative values in $\{c_{i,m} Y_i X_{ij}^{-1} : i = 1, \ldots, n\}$ as $0 = \xi_0 > \xi_1 > \xi_2 > \cdots$; set $b = 0$, $l = 0$, $T = F'(0-)$; while $T > 0$, update $T \mapsto T - \sum_{i: c_{i,m} Y_i X_{ij}^{-1} = \xi_{l+1}} |X_{ij}|$, $b \mapsto \xi_{l+1}$, $l \mapsto l + 1$; upon stopping set $\hat{\beta}_{j,m} = \xi_l$.

   (iv) Else if $F'(0+) < 0$, order all positive values in $\{c_{i,m} Y_i X_{ij}^{-1} : i = 1, \ldots, n\}$ as $0 = \xi_0 < \xi_1 < \xi_2 < \cdots$; set $b = 0$, $l = 0$, $T = F'(0+)$; while $T < 0$, update $T \mapsto T + \sum_{l: c_{i,m} Y_i X_{ij}^{-1} = \xi_{l+1}} |X_{ij}|$, $b \mapsto \xi_{l+1}$, $l \mapsto l + 1$; upon stopping set $\hat{\beta}_{j,m} = \xi_l$.

4. Selection step: Compute $\sum_{i=1}^{n} (c_{i,m} - Y_i X_{ij} \hat{\beta}_{j,m})_+$ for $j = 1, \ldots, d$, and find

$$j_m^* = \arg_j \min \sum_{i=1}^{n} (c_{i,m} - Y_i X_{ij} \hat{\beta}_{j,m})_+. \tag{16}$$

   Compute the error $S_m = \sum_{i=1}^{n} (c_{i,m} - Y_i X_{ij_m^*} \hat{\beta}_{j_m^*,m})_+$.

5. Stopping rule: If $S_{m-1} - S_m < \epsilon$, stop; else update $c_{i,m+1} \mapsto c_{i,m} - Y_i X_{ij_m^*,m} \hat{\beta}_{j_m^*,m}$ and $m \mapsto m + 1$, and go back to the optimization step 3.

6. Prediction rule: Classify new observation with predictors $\boldsymbol{X} = (X_1, \ldots, X_d)$ to group 1 if

$$\hat{\beta}_0 + \sum_{k=1}^{m} \hat{\beta}_{j_k^*,k} X_{j_k^*} \geq 0.$$

The choice of the tuning parameter λ is extremely important. A range of possible values of λ should be identified and divided into a grid, not necessarily with equal-spaced points. The algorithm is repeated over all possible values of λ. Based on a separate segment of the data not already used in estimation (known as the validation set), the misclassification rate is calculated for all values of λ on the chosen grid. The value of λ leading to the smallest misclassification rate is chosen. In practice, a k-fold cross validation is often used as an alternative tuning procedure, and the value $k = 5$ is a typical choice. The final procedure will be called the CLASsification and Selection using Iterative Cycles (CLASSIC). The aggressive version will be called aggressive CLASSIC1 (agCLASSIC1).

A small variation of CLASSIC is obtained by performing the intercept correction step after each variable selection step. Abbreviating c_i for $c_{i,m}$, the objective function

$$\begin{aligned} F(b) &= \sum_{i:Y_i=1} (c_i - b)_+ + \sum_{i:Y_i=1} (c_i + b)_+ \\ &= \sum_{i:Y_i=1, c_i \geq b} c_i + \sum_{i:Y_i=-1, c_i \geq -b} c_i + b(\#\{i : Y_i = -1, c_i \geq -b\} - \#\{i : Y_i = 1, c_i \geq b\}). \end{aligned}$$

The function is differentiable except at knot points $\{c_i Y_i : i = 1, \ldots, n\}$. For any point b, the right and left derivatives are given by

$$\begin{aligned} F'(b+) &= \#\{i : Y_i = -1, c_i \geq -b\} - \#\{i : Y_i = 1, c_i > b\}, \\ F'(b-) &= \#\{i : Y_i = -1, c_i > -b\} - \#\{i : Y_i = 1, c_i \geq b\}. \end{aligned}$$

This follows by specializing (10) and (11) to the case with $X_{ij} = 1$ for all $i = 1, \ldots, n$ and $\lambda = 0$. Under the assumption that $N_+, N_- > 0$, we have $F'(-\infty) = -N_+ < 0$ and $F'(\infty) = N_- > 0$, so $F(b)$ has a minima. The search for the minima will proceed as in the case of other predictors.

- If $F'(0-) \leq 0 \leq F'(0+)$, zero is the minima.
- If $F'(0-) > 0$, order all negative values in $\{c_i Y_i : i = 1, \ldots, n\}$ as $0 = \xi_0 > \xi_1 > \xi_2 > \cdots$; set $b = 0$, $l = 0$, $T = F'(0-)$; while $T > 0$, update T to $T - \#\{i : c_i Y_i = \xi_{l+1}\}$, b to ξ_{l+1}, l to $l + 1$; upon stopping set the minima to ξ_l.
- If $F'(0+) < 0$, order all positive values in $\{c_i Y_i : i = 1, \ldots, n\}$ as $0 = \xi_0 < \xi_1 < \xi_2 < \cdots$; set $b = 0$, $l = 0$, $T = F'(0+)$; while $T < 0$, update T to $T + \#\{i : c_{i,m} Y_i = \xi_{l+1}\}$, b to ξ_{l+1}, l to $l + 1$; upon stopping set the minima to ξ_l.

The resulting minima is the correction which needs to be added to the current value of the intercept. Note that in a linear regression model with centered variables, such a step would be redundant for FIRST. For classification, non-zero values of corrections are possible, but typically the amount of correction is small.

3 Extensions

Since the objective functions in (6) and (8) are not differentiable, locating the minima involves calculating right and left derivatives separately. This may be avoided by considering an analogous

objective function, which is differentiable at the same time, by replacing power 1 by power 2. We shall refer the former as CLASSIC1 and the latter as CLASSIC2. The squared hinge loss is related to the proximal SVMs studied in Fung and Mangasarian (2001). Consider, for the intercept term, the minimization of the function

$$\sum_{i=1}^{n}(1-Y_ib)_+^2 = (1-b)_+^2 N_+ + (1+b)_+^2 N_- \tag{17}$$

with respect to b, and for the jth predictor, $j = 1, \ldots, p$, the minimization of the function

$$F(b) = \sum_{i=1}^{n}(c_i - Y_i b X_{ij})_+^2 + \lambda|b_j|. \tag{18}$$

The minimizer of (17) is given by

$$b_0 = (N_+ - N_-)/n, \tag{19}$$

assuming that $N_+ > 0$ and $N_- > 0$. The proof of (19) is given in the appendix.

Once b_0 is chosen, c_i is updated to $c_i - Y_i b_0$. With this new c_i, now b_j is chosen as the minimizer of (18). Since the resulting function F is differentiable except at zero, convex and piecewise quadratic, the resulting solution is $b_j = 0$ if $F'(0-) \le 0 \le F'(0+)$ or is given by the unique solution of $F'(b) = 0$ otherwise.

For any $b \neq 0$, $F'(b)$ can be calculated as

$$\begin{aligned} F'(b) &= -2\sum_{i:c_i > Y_i X_{ij} b} c_i Y_i X_{ij} + 2b \sum_{i:c_i > Y_i X_{ij} b} X_{ij}^2 + \lambda \text{sign}(b) \\ &= 2(-T_1 - T_2 + T_3 + T_4) + 2b(T_5 + T_6 + T_7 + T_8) + \lambda \text{sign}(b), \end{aligned} \tag{20}$$

where

$$\begin{aligned} T_1 &= \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1} > b} c_i X_{ij}, \qquad T_5 = \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1} > b} X_{ij}^2, \\ T_2 &= \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1} < b} c_i X_{ij}, \qquad T_6 = \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1} < b} X_{ij}^2, \\ T_3 &= \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1} > -b} c_i X_{ij}, \qquad T_7 = \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1} > -b} X_{ij}^2, \\ T_4 &= \sum_{i:Y_i=-1, X_{ij}<0, c_i X_{ij}^{-1} < -b} c_i X_{ij}, \qquad T_8 = \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1} < -b} X_{ij}^2. \end{aligned}$$

In particular, letting $b \downarrow 0$ and $b \uparrow 0$ respectively, we obtain

$$F'(0+) = 2(-T_1^0 - T_2^0 + T_3^0 + T_4^0) + \lambda, \tag{21}$$

$$F'(0-) = 2(-T_1^0 - T_2^0 + T_3^0 + T_4^0) - \lambda, \tag{22}$$

where

$$T_1^0 = \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1}>0} c_i X_{ij}, \qquad T_2^0 = \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1}<0} c_i X_{ij},$$

$$T_3^0 = \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1}>0} c_i X_{ij}, \qquad T_4^0 = \sum_{i:Y_i=-1, X_{ij}<0, c_i X_{ij}^{-1}<0} c_i X_{ij}.$$

Note that $F'(0-) \leq 0 \leq F'(0+)$ if and only if $|-T_1^0 - T_2^0 + T_3^0 + T_4^0| \leq \lambda/2$.

Since $F'(b)$ is piecewise linear, increasing and continuous (except at $b = 0$) with knot points $\{c_i Y_i X_{ij}^{-1} : i = 1, \ldots, n\}$, the solution of $F'(b) = 0$ may be found by evaluating $F'(b)$ at all knot points, locating the interval where the solution lies and finally finding the precise point by linear interpolation. As in the case of CLASSIC1, the naive computation of (20) at all knot points from scratch involves $O(n^2)$ steps. As before, we need to compute the derivative values only at knot points that are relevant for locating the solution, and exploit a recurrence relation between the values of $F'(b)$ at neighboring knots. Then the evaluations of $F'(b)$ can proceed from $b = 0$ in the direction of the solution until the knot-interval containing the solution is located. This leads to a reduction of $O(n^2)$ computations to only $O(n)$, as before.

To describe the recurrence relation, order all knots $\{c_i Y_i X_{ij}^{-1} : i = 1, \ldots, n\}$. If ξ is a negative knot with $F'(\xi)$ given by (20), then for ξ^* the knot on its left closest to ξ, $F'(\xi^*)$ can be computed by updating $T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8$ respectively to

$$T_1^* = T_1 + \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1}=\xi} c_i X_{ij}, \qquad T_5^* = T_5 + \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1}=\xi} X_{ij}^2, \tag{23}$$

$$T_2^* = T_2 - \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1}=\xi^*} c_i X_{ij}, \qquad T_6^* = T_6 - \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1}=\xi^*} X_{ij}^2, \tag{24}$$

$$T_3^* = T_3 - \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1}=-\xi^*} c_i X_{ij}, \qquad T_7^* = T_7 - \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1}=-\xi^*} X_{ij}^2, \tag{25}$$

$$T_4^* = T_4 + \sum_{i:Y_i=-1, X_{ij}<0, c_i X_{ij}^{-1}=-\xi} c_i X_{ij}, \qquad T_8^* = T_8 + \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1}=-\xi} X_{ij}^2. \tag{26}$$

In the above updating formulas, $\xi = 0$ is allowed with the interpretation that F' stands for the left hand derivative $F'(0-)$ given by (22) and the initial values of T_5, T_6, T_7, T_8 are given by

$$T_5^0 = \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1} \geq 0} X_{ij}^2, \quad T_6^0 = \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1} < 0} X_{ij}^2,$$
$$T_7^0 = \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1} > 0} X_{ij}^2, \quad T_8^0 = \sum_{i:Y_i=-1, X_{ij}<0, c_i X_{ij}^{-1} \leq 0} X_{ij}^2.$$

On the other hand, if ξ is a positive knot with $T = F'(\xi)$ given by (20), then for ξ^* the knot on its right closest to ξ, $F'(\xi^*)$ can be computed by updating $T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8$ respectively to

$$T_1^* = T_1 - \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1} = \xi^*} c_i X_{ij}, \quad T_5^* = T_5 - \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1} = \xi^*} X_{ij}^2, \tag{27}$$
$$T_2^* = T_2 + \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1} = \xi} c_i X_{ij}, \quad T_6^* = T_6 + \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1} = \xi} X_{ij}^2, \tag{28}$$
$$T_3^* = T_3 + \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1} = -\xi} c_i X_{ij}, \quad T_7^* = T_7 + \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1} = -\xi} X_{ij}^2, \tag{29}$$
$$T_4^* = T_4 - \sum_{i:Y_i=-1, X_{ij}<0, c_i X_{ij}^{-1} = -\xi^*} c_i X_{ij}, \quad T_8^* = T_8 - \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1} = -\xi^*} X_{ij}^2. \tag{30}$$

In the above updating formula, $\xi = 0$ is allowed with the interpretation that F' stands for the right hand derivative $F'(0+)$ given by (21) and and the initial values of T_5, T_6, T_7, T_8 are given by

$$T_5^0 = \sum_{i:Y_i=1, X_{ij}>0, c_i X_{ij}^{-1} > 0} X_{ij}^2, \quad T_6^0 = \sum_{i:Y_i=1, X_{ij}<0, c_i X_{ij}^{-1} \leq 0} X_{ij}^2,$$
$$T_7^0 = \sum_{i:Y_i=-1, X_{ij}>0, c_i X_{ij}^{-1} \geq 0} X_{ij}^2, \quad T_8^0 = \sum_{i:Y_i=-1, X_{ij}<0, c_i X_{ij}^{-1} < 0} X_{ij}^2.$$

The updating formulas (23)–(30) follow from calculations as in the case of CLASSIC1 provided in the appendix.

Finally to locate the actual solution, first we locate the interval (ξ^*, ξ) on the negative side if $F'(0-) > 0$ or (ξ, ξ^*) on the positive side if $F'(0+) > 0$. In either case, linear interpolation gives the root for $F'(b) = 0$ through the equation

$$\frac{b - \xi}{0 - F'(\xi)} = \frac{\xi^* - \xi}{F'(\xi^*) - F'(\xi)}, \quad \text{or} \quad b = \xi - F'(\xi) \frac{\xi - \xi^*}{F'(\xi) - F'(\xi^*)}. \tag{31}$$

The whole procedure can be described by Algorithm 2. We first fix a precision parameter $\epsilon > 0$. For any given value of the tuning parameter $\lambda > 0$, do the following steps:

Algorithm 2

1. Standardization step: Replace X_{ij} by $(X_{ij} - \bar{X}_j)/\{\sum_{i=1}^n (X_{ij} - \bar{X}_j)^2\}^{1/2}$ for $i = 1, \ldots, n$, and $j = 1, \ldots, d$, where $\bar{X}_j = n^{-1} \sum_{i=1}^n X_{ij}$. Thus from now on, we shall assume that $\bar{X}_j = 0$ and $\sum_{i=1}^n X_{ij}^2 = 1$.

2. Intercept estimation step: Compute $N_+ = \#\{i : Y_i = 1\}$ and $N_- = \#\{i : Y_i = -1\}$ and set $\hat{\beta}_0 = (N_- - N_+)/n$. Compute $S_0 = \sum_{i=1}^n (1 - Y_i \hat{\beta}_0)_+^2$ and set $m = 1$ and $c_{i,m} = 1 - Y_i \hat{\beta}_0$.

3. Optimization step: For each $j = 1, \ldots, d$, do the following:

   (i) Compute $T_1^0, T_2^0, T_3^0, T_4^0$.

   (ii) If $| - T_1^0 - T_2^0 + T_3^0 + T_4^0| \le \lambda$, set $\hat{\beta}_{j,m} = 0$ and stop. [For the aggressive version, permanently remove j from the pool of potential predictors.]

   (iii) Else compute $F'(0-)$ and $F'(0+)$ respectively by (21) and (22).

   (iv) If $F'(0-) > 0$, order all negative values in $\{c_{i,m} Y_i X_{ij}^{-1} : i = 1, \ldots, n\}$ as $0 = \xi_0 > \xi_1 > \xi_2 > \cdots$; set $b = 0$, $l = 0$, $T = F'(0-)$; while $T > 0$, update $T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8$ respectively to $T_1^*, T_2^*, T_3^*, T_4^*, T_5^*, T_6^*, T_7^*, T_8^*$ using (23)–(26), $b \mapsto \xi_{l+1}$, $l \mapsto l + 1$.

   (v) Else if $F'(0+) < 0$, order all positive values in $\{c_{i,m} Y_i X_{ij}^{-1} : i = 1, \ldots, n\}$ as $0 = \xi_0 < \xi_1 < \xi_2 < \cdots$; set $b = 0$, $l = 0$, $T = F'(0+)$; while $T < 0$, update $T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8$ respectively to $T_1^*, T_2^*, T_3^*, T_4^*, T_5^*, T_6^*, T_7^*, T_8^*$ using (27)–(30), $b \mapsto \xi_{l+1}$, $l \mapsto l + 1$.

   (vi) Upon stopping the recursive step, locate the optimizer by

$$b = \xi_{l-1} - F'(\xi_{l-1}) \frac{\xi_{l-1} - \xi_l}{F'(\xi_{l-1}) - F'(\xi_l)}.$$

4. Selection step: Compute $\sum_{i=1}^n (c_{i,m} - Y_i X_{ij} \hat{\beta}_{j,m})_+^2$ for $j = 1, \ldots, d$, and find

$$j_m^* = \arg_j \min \sum_{i=1}^n (c_{i,m} - Y_i X_{ij} \hat{\beta}_{j,m})_+^2. \tag{32}$$

   Compute the error $S_m = \sum_{i=1}^n (c_{i,m} - Y_i X_{ij_m^*} \hat{\beta}_{j_m^*,m})_+^2$.

5. Stopping rule: If $S_{m-1} - S_m < \epsilon$, stop; else update $c_{i,m+1} \mapsto c_{i,m} - Y_i X_{ij^*_m,m} \hat{\beta}_{j^*_m,m}$ and $m \to m+1$, and go back to the optimization step.

6. Prediction rule: Classify new observation with predictors $\boldsymbol{X} = (X_1, \ldots, X_d)$ to group 1 if

$$\hat{\beta}_0 + \sum_{k=1}^{m} \hat{\beta}_{j^*_k,k} X_{j^*_k} \geq 0.$$

The choice of the tuning parameter λ is again extremely important. As before, the optimizing value of λ on a grid (not necessarily with equispaced points) may be found by k-fold cross validation for some suitable k.

Also, as in the case with power 1 (i.e. CLASSIC1), a variation is obtained by performing the intercept correction step after each variable selection step. Abbreviating c_i for $c_{i,m}$, the objective function $F(b) = \sum_{i=1}^{n} (c_i - Y_i b)_+^2$. The function is differentiable everywhere with derivative given by

$$\begin{aligned} F'(b) &= -2 \sum_{i:Y_i=1, c_i>b} c_i + 2 \sum_{i:Y_i=-1, c_i>b} c_i + 2b(\#\{i : Y_i = 1, c_i > b\} + \#\{i : Y_i = -1, c_i > -b\}) \\ &= 2(-J_1 + J_2) + 2b(J_3 + J_4), \end{aligned}$$

say. In particular, for $b = 0$, $J_1 = \sum_{i:Y_i=1, c_i>0} c_i$, $J_2 = \sum_{i:Y_i=-1, c_i>0} c_i$, $J_3 = J_4 = 0$. The location of the root proceeds in a similar way. If $J_1 = J_2$, $b = 0$ is the solution. If $J_1 < J_2$, the root is located on the left side. Order all knots $\{c_i : Y_i = -1\}$ as $0 = \xi_0 > \xi_1 > \cdots$; set $b = 0$, $l = 0$, $T = F'(b)$; while $F'(b) > 0$, update J_1 to $J_1^* = J_1 + \xi_l \#\{i : Y_i = 1, c_i = \xi_l\}$, $J_2^* = J_2 + \xi_{l+1} \#\{i : Y_i = -1, c_i = -\xi_{l+1}\}$, $J_3^* = J_3 + \#\{i : Y_i = 1, c_i = \xi_l\}$, $J_4^* = J_4 + \#\{i : Y_i = -1, c_i = -\xi_{l+1}\}$.

3.1 Basic Properties

The criterion function of CLASSIC with power 1 or 2 (and their intercept corrected versions) decreases in every iteration decreases. This happens since by the definition of the procedure, $\sum_{i=1}^{n} (c_{i,m-1} - Y_i X_{ij} \hat{\beta}_{j^*_m,m})_+^p \leq \sum_{i=1}^{n} (c_{i,m-1} - Y_i X_{ij} 0)_+^p$, $p = 1, 2$, as zero is always a possible value of the coefficient, and the left hand side is the criterion function at stage m and the right hand side is the criterion function at the $(m-1)$th stage. In particular, this ensures that the algorithm converges within n/ϵ steps, where ϵ is the accuracy level chosen in the stopping rule, since every improvement has to be at least ϵ.

3.2 Relations with Existing Methods

Our proposed method share some properties with boosting methods. Boosting is an iterative approach for building sparse models, and has been developed for linear regression models. Boosting can be regarded as a gradient descent method in function space, and hence can be regarded as a regularization scheme for model estimation; see Breiman (1998), Mason et al. (1999), and Friedman et al. (2000). Buhlmann (2004) considered boosting with the squared error loss, called L_2-boosting. They showed that L_2-boosting for linear models produces consistent estimates for high dimensional linear models, provided that some appropriate stopping criterion is used. Efron et al. (2004) pointed out a link between L_2-boosting and the LASSO. The tuning parameter in L_2-boosting is the number of boosting iterations, and can be selected by the Akaike Information Criterion. Boosting in the context of classification data has been reviewed in Freund, Schapire and Abe (1999); see also Gao et al. (2006) in this context.

The co-ordinatewise optimization idea of the proposed method has some similarity in spirit with the coordinate-wise descent algorithm (CDA) proposed by Friedman et al. (2007) as an alternative to the LARS algorithm for LASSO in linear models. The main idea of the coordinate-wise descent algorithm is to successively minimize the objective function with respect to one parameter at a time, while holding the remaining parameters at their current values. The main difference between the current approach and the CDA is that in the former, only one coefficient corresponding to the best predictor is updated while in CDA, all coefficients are updated one at a time in each iteration. As a result, the proposed procedure gives a new statistical method, while the CDA, if it converges, gives an algorithm for computing the global minimizer of the objective function (i.e., LASSO for the linear model).

4 Simulation Examples

We use simulations to compare our algorithms with existing variable selection algorithms in various classification problems. To test the variable selection ability of the proposed method, we consider sparse settings with very few important predictors and a lot of redundant predictors in the simulation models.

All simulations are run on the cluster at the Department of Statistics of North Carolina State University. We compare the proposed methods CLASSIC with powers 1 and 2 and their aggressive versions, denoted agCLASSIC1 and agCLASSIC2, with several competing variable selection classi-

fication methods. We implement the logistic regression method with lasso and elastic net penalty described in Friedman et al. (2010), denoted GLM-Lasso and GLM-ENET respectively. We also compare the hybrid huberized support vector machine method (HH-SVM) outlined in Wang et al. (2008) and its generalizations with squared hinge loss function (SQ-SVM) and logistic regression loss function (Logit-SVM). Lastly, we include in our simulation the L1-SVM method developed by Zou (2007) as well as the popular SCAD-SVM and regular SVM methods.

All these existing methods are implemented using their existing packages in R. In particular, GLM-Lasso and GLM-ENET are implemented using the "glmnet" package, and HH-SVM, SQ-SVM, and Logit-SVM are implemented using the "gcdnet" package. L1-SVM and SCAD-SVM are executed using the "penalizedSVM" package, and lastly SVM is implemented using the package "svmpath".

We implement all the CLASSIC methods with $\epsilon = 10^{-8}$. To tune λ, we use an independent validation set of the same sample size n as the training set and choose λ to give the minimum tuning error. Then we evaluate the classification accuracy of the procedure on $n' = 20n$ test data points by computing the test error, which is the misclassification rate. The competing methods are trained and tested in the exact same manner. To measure selection accuracy, we consider two types of error: number of false negatives which is defined as the number of non-zero coefficients which are estimated as zero, and number of false positives, which is defined as the number of zero coefficients which are not estimated as zero. We record test error, false negative rate, false positive rate, and total computing time for each procedure based on 1000 replications in low and moderate dimensional examples, with the exception of setting S3 which uses 100 replications, while we use 100 replications in the high and ultra dimensional case. All entries reported are average based on all replications. We also report the performance of the Bayes rule based on the correct choice of variables as the gold standard (Oracle).

4.1 Low dimension, $n > d$, independent covariates

We run 1000 simulations for $n = 50$. The data $(\boldsymbol{X}_i, Y_i)$ are generated as

$$\mathrm{P}(Y_i = 1|\boldsymbol{X}_i) = \exp(1 + \boldsymbol{X}_i^T\boldsymbol{\beta})/\{1 + \exp(1 + \boldsymbol{X}_i^T\boldsymbol{\beta})\},$$

with $\boldsymbol{X} = (X_1, ..., X_d)^T$ following the multivariate normal distribution with mean $\boldsymbol{\mu} = \mathbf{0}$ and covariance $\Sigma = \mathbf{I}_d$. The true coefficient vector $\boldsymbol{\beta} = (3, 3, 0, \cdots, 0)^T$, and the number of nonzero

coefficients in β is $d_0 = 2$. We consider two dimension settings $d = 8$ and $d = 20$. The results are summarized in Table 1. We observe that in both cases at least one of the proposed CLASSIC methods outperforms the field of competitors in terms of test error. The L1-SVM method suffered from numerical instabilities and was unable to consistently provide results without crashing for each low dimensional setting so its results are not reported. The false negative and false positive rates are comparable accross all the methods with the exception of agCLASSIC1, which selects a sparser model than its competitors resulting in a lower false positive rate. The computation time for all the CLASSIC methods, with the exception of CLASSIC2, are less than that of the popular SCAD-SVM.

Table 1: Classification and selection results for low-dimensional independent predictors

Method	Test Error		False Negative		False Positive		Time	
	$d = 8$	$d = 20$	$d = 8$	$d = 20$	$d = 8$	$d = 20$	$d = 8$	$d = 20$
CLASSIC1	0.161	0.184	0.00	0.01	3.63	8.75	0.20	0.52
CLASSIC2	0.152	0.165	0.00	0.00	3.37	6.18	1.36	2.57
agCLASSIC1	0.155	0.158	0.08	0.07	0.39	0.86	0.13	0.14
agCLASSIC2	0.154	0.170	0.00	0.00	3.20	6.19	0.19	0.23
GLM-Lasso	0.151	0.161	0.00	0.00	2.64	5.15	0.09	0.09
GLM-ENET	0.155	0.173	0.00	0.00	3.63	8.24	0.09	0.09
HH-SVM	0.197	0.215	0.00	0.00	2.89	9.47	0.01	0.01
Logit-SVM	0.245	0.258	0.00	0.00	2.85	9.43	0.01	0.02
SQ-SVM	0.167	0.208	0.00	0.00	4.97	15.06	0.01	0.01
SCAD-SVM	0.152	0.169	0.00	0.03	2.16	3.10	0.75	0.39
SVM	0.183	0.249	-	-	-	-	0.03	0.04
Oracle	0.143	0.143	-	-	-	-	-	-

4.2 Moderate dimension, comparable n and d, dependent covariates

In this subsection we evaluate the performance of the CLASSIC methods when the dimension d is moderately large and its magnitude is comparable with the sample size n. The data $(\boldsymbol{X}_i, Y_i)$ are

generated as following the model

$$\mathrm{P}(Y_i = 1|\boldsymbol{X}_i) = \Phi(\boldsymbol{X}_i^T\boldsymbol{\beta})), \tag{33}$$

where Φ is the cumulative distribution function of $N(0,1)$. The covariance matrix of $\boldsymbol{X}$ has AR(1) structure, $\mathrm{corr}(X_i, X_j) = 0.5^{|i-j|}$ for $i \neq j$. The experiments are conducted under the following three scenarios:

- Setting 1 (S1): $n = 50$, $d = 50$, $d_0 = 5$, $\beta_1, \cdots, \beta_d \overset{\text{iid}}{\sim} \mathrm{Unif}(0.4, 1.2)$;
- Setting 2 (S2): $n = 100$, $d = 100$, $d_0 = 12$, $\beta_1, \cdots, \beta_d \overset{\text{iid}}{\sim} \mathrm{Unif}(0, 0.5)$;
- Setting 3 (S3): $n = 200$, $d = 200$, $d_0 = 20$, $\beta_1, \cdots, \beta_d \overset{\text{iid}}{\sim} \mathrm{Unif}(0, 0.4)$.

Table 2 suggests that, under all the three settings (S1), (S2), (S3), the new CLASSIC methods perform better or comparably with the competing methods with the exception of GLM-Lasso and GLM-ENET.

Table 2: Classification and selection results for moderate dimension

Method	Test Error			False Negative			False Positive			Time		
	(S1)	(S2)	(S3)	(S1)	(S2)	(S3)	(S1)	(S2)	(S3)	(S1)	(S2)	(S3)
CLASSIC1	0.214	0.281	0.272	1.07	3.32	5.21	13.64	43.34	85.76	0.64	21.73	1311.96
CLASSIC2	0.195	0.257	0.243	0.80	4.51	7.78	12.68	21.63	29.78	0.77	10.41	82.42
agCLASSIC1	0.183	0.261	0.282	1.10	6.74	14.16	1.62	1.30	0.08	0.15	0.34	0.53
agCLASSIC2	0.197	0.260	0.243	0.65	3.79	7.05	17.97	29.61	33.87	0.41	2.58	18.23
GLM-Lasso	0.182	0.246	0.230	0.62	4.14	6.29	7.34	12.96	21.98	0.10	0.11	0.13
GLM-ENET	0.178	0.239	0.225	0.17	2.80	4.65	12.66	18.19	27.85	0.10	0.11	0.13
L1-SVM	0.201	-	-	0.96	-	-	10.36	-	-	2.31	-	-
HH-SVM	0.193	0.244	0.226	0.04	1.43	2.17	19.01	28.36	37.78	0.01	0.02	0.02
Logit-SVM	0.212	0.253	0.233	0.02	0.94	1.33	20.86	31.10	41.40	0.04	0.12	0.28
SQ-SVM	0.210	0.265	0.248	0.04	0.99	1.65	31.43	57.20	102.90	0.01	0.02	0.03
SCAD-SVM	0.237	0.296	0.261	1.46	5.88	12.21	10.63	20.30	14.22	0.45	1.35	5.33
SVM	0.243	0.288	0.284	-	-	-	-	-	-	0.03	0.09	0.34
Oracle	0.162	0.250	0.243	-	-	-	-	-	-	-	-	-

4.3 High dimension, $n < d$, dependent covariates

In this subsection, we present results for the situation when the dimension is very high but the sample size is much smaller compared to the dimension. As in the previous example, the covariates X have AR(1) structure among all the components, $\text{corr}(X_i, X_j) = 0.5^{|i-j|}$ for $i \neq j$ and the dependent binary variable Y is generated through probit regression as in (33) We consider the following two scenarios:

- Setting 1 (S1)$'$: $n = 50$, $d = 500$, $d_0 = 12$, $\beta_1, \cdots, \beta_d \overset{\text{iid}}{\sim} \text{Unif}(0, 0.5)$;
- Setting 2 (S2)$'$: $n = 50$, $d = 1000$, $d_0 = 12$, $\beta_1, \cdots, \beta_d \overset{\text{iid}}{\sim} \text{Unif}(0, 0.5)$.

Table 3 shows that the CLASSIC methods outperform or nearly outperform competing methods in terms of test error while offering the lowest false positive rates. The CLASSIC methods are also able to offer competitive computational times for these high dimensional data sets. In particular, the agCLASSIC1 method selects a sparser model than the popular L1-SVM and SCAD-SVM methods while offering a lower test error and at least six times faster computation time.

Table 3: Classification and selection results for high dimension

Method	Test Error		False Negative		False Positive		Time	
	(S1)$'$	(S2)$'$	(S1)$'$	(S2)$'$	(S1)$'$	(S2)$'$	(S1)$'$	(S2)$'$
CLASSIC1	0.382	0.403	9.51	10.16	12.85	11.35	1.37	2.24
CLASSIC2	0.350	0.368	8.97	9.68	12.28	10.61	2.00	3.45
agCLASSIC1	0.361	0.384	9.19	9.79	5.89	6.64	0.29	0.55
agCLASSIC2	0.351	0.367	9.01	9.68	11.98	10.37	1.67	2.96
GLM-Lasso	0.348	0.368	7.87	8.48	15.59	17.06	0.10	0.15
GLM-ENET	0.346	0.369	5.61	6.13	40.09	50.47	0.10	0.15
L1-SVM	0.373	0.393	7.26	7.59	60.71	106.47	2.06	3.34
HH-SVM	0.363	0.385	4.37	5.02	69.92	95.24	0.03	0.05
Logit-SVM	0.374	0.397	3.43	3.91	96.54	145.64	0.23	0.80
SQ-SVM	0.364	0.385	3.57	4.32	109.57	136.57	0.03	0.04
SCAD-SVM	0.368	0.386	6.47	6.74	104.22	183.80	20.70	6.18
SVM	0.415	0.440	-	-	-	-	0.02	0.03
Oracle	0.251	0.245	-	-	-	-	-	-

4.4 Ultra high dimension, $n \ll d$, dependent covariates

We also explore two scenarios where the number of predictors is vastly greater than the sample size. The covariates again follow an AR(1) structure, $\text{corr}(X_i, X_j) = 0.5^{|i-j|}$ for $i \neq j$, and the dependent variable, Y, is generated from probit regression as in (33).

- Setting 1 (S1)$''$: $n = 200$, $d = 10000$, $d_0 = 12$, $\beta_1, \cdots, \beta_d \overset{\text{iid}}{\sim} \text{Unif}(0, 0.5)$;
- Setting 2 (S2)$''$: $n = 400$, $d = 20000$, $d_0 = 12$, $\beta_1, \cdots, \beta_d \overset{\text{iid}}{\sim} \text{Unif}(0, 0.5)$.

Due to the intensely large dimension of these settings we only implement the aggressive CLASSIC methods, GLM-Lasso, L1-SVM, and SVM. It is worth noting that L1-SVM was unable to consistently run to completion for the (S2)$''$ setting so it does not have results to report for that setting.

Table 4 shows that the agCLASSIC1 method offers the best combination of low test errors and low false negative and false positive rates. These settings also show that the agCLASSIC1 method can be completed in a reasonable time frame for ultra high dimensional cases.

Table 4: Classification and selection results for ultra high dimension

Method	Test Error		False Negative		False Positive		Time	
	(S3)$'$	(S4)$'$	(S3)$'$	(S4)$'$	(S3)$'$	(S4)$'$	(S3)$'$	(S4)$'$
agCLASSIC1	0.247	0.221	6.34	5.54	2.12	2.36	20.53	168.62
agCLASSIC2	0.266	0.227	7.54	4.88	11.94	21.28	419.28	6168.49
GLM-Lasso	0.247	0.211	5.42	3.56	26.78	41.60	0.856	3.06
L1-SVM	0.314	-	4.58	-	197.92	-	288.56	-
SVM	0.463	0.477	-	-	-	-	1.53	14.37
Oracle	0.250	0.248	-	-	-	-	-	-

4.5 Convergence diagnostics

Table 5 displays the average number of iterations till convergence for each of the previously defined simulation settings for each of the CLASSIC methods. It can be seen that the agCLASSIC1 and agCLASSIC2 methods converge rather quickly for all the simulation settings. CLASSIC1 and CLASSIC2, however, require a considerably larger number of iterations for the moderate dimensional cases but converge quickly for the high dimensional cases.

Table 5: Average number of iterations till convergence for all simulation settings, standard errors are displayed in parentheses

Method	$d = 8$	$d = 20$	(S1)	(S2)	(S3)	(S1)′	(S2)′	(S3)′	(S4)′
CLASSIC1	48.43	94.04	87.84	759.49	25911.12	36.32	24.11	-	-
	(2.20)	(4.56)	(4.86)	(38.54)	(5582.64)	(3.31)	(1.35)	(-)	(-)
CLASSIC2	443.10	103.68	42.35	92.90	129.99	17.68	14.75	-	-
	(111.70)	(13.98)	(1.39)	(4.22)	(21.93)	(0.28)	(0.23)	(-)	(-)
agCLASSIC1	3.44	3.95	6.78	7.93	8.67	9.91	9.97	8.98	10.52
	(0.035)	(0.06)	(0.09)	(0.16)	(0.87)	(0.14)	(0.13)	(0.79)	(1.64)
agCLASSIC2	167.84	16.10	34.23	56.72	56.26	16.93	14.33	18.06	30.88
	(72.60)	(1.77)	(0.90)	(1.76)	(7.00)	(0.26)	(0.22)	(2.33)	(3.38)

5 Real Data

We use two microarray gene expression data sets to illustrate the performance of the proposed CLASSIC algorithms on real-world problems. The first data set we use is a colon cancer dataset (Alon et al. , 1999). The dataset contains information of 62 samples on 2000 genes. The samples belong to tumor and normal colon tissues. The data is available at `http://microarray.princeton.edu/oncology`. We randomly divide the data into three equal parts, with the first two parts used as the training and tuning sets and the last part as the test set. This procedure is repeated 100 times, and the average test error, number of genes selected in the final model, and computation time is reported in Table 6. The results show that the CLASSIC methods give a promising performance, offering competitive test errors while selecting a significantly sparser model than competing methods.

Table 6: Misclassification rates of classification rules for the colon cancer data

	Test Error	Num. Genes Selected	Time
Classic1	0.258	5.07	0.462
Classic2	0.249	6.32	0.944
agClassic1	0.250	4.13	0.200
agClassic2	0.251	6.18	0.829
GLM-Lasso	0.245	11.41	0.112
GLM-ENET	0.219	54.73	0.116
L1-SVM	0.217	146.35	1.674
HH-SVM	0.220	99.12	0.019
Logit-SVM	0.233	150.45	0.212
SQ-SVM	0.221	96.85	0.015
SCAD	0.245	430.30	6.911
SVM	0.266	2000.00	0.018

The second data set we explore is prostate cancer data featured in Singh et al. (2002). The data set is available at `http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi`. It contains 102 patient samples of 12600 genes with 52 of the subjects having prostate tumor samples and the remaining 50 having normal prostate samples. The data is randomly divided into three equal parts with training, tuning, and testing executed in an identical manner as the colon cancer data example. The randomization process is again replicated 100 times. Table 7 shows that each of the CLASSIC methods is able to handle the large gene expression data set in a reasonable amount of time. Our proposed methods provide promising test errors using fewer genes than competing methods.

Table 7: Misclassification rates, number of genes selected, and computing time for prostate cancer data

	Test Error	Num. Genes Selected	Time
Classic1	0.141	6.41	5.542
Classic2	0.150	5.93	8.336
agClassic1	0.138	4.37	2.703
agClassic2	0.150	5.88	7.321
GLM-Lasso	0.120	16.76	0.289
GLM-ENET	0.101	76.76	0.291
L1 SVM	0.106	399.34	17.275
HH-SVM	0.101	132.94	0.208
Logit-SVM	0.101	331.94	4.838
SQ-SVM	0.097	134.09	0.179
SCAD	0.104	1476.52	32.179
SVM	0.112	12600.00	0.065

6 Discussion

We suggest a new variable selection method called CLASSIC for high dimensional sparse classification problems using a recursive algorithm that iterates over possible predictors. The new technique CLASSIC is a combination of one-dimensional ℓ_1-penalized support vector machines and forward selection, and comes in different variations. The main advantage of the proposed approach is that the optimizer in one dimension can be located very easily using a simple iterative formula which terminates optimally. Then the best predictor can be selected and the procedure will continue following a forward selection approach. The resulting procedure terminates within a fixed number of steps, and typically much more quickly.

The iterative approach to finding the best sparse linear combination of predictors in classifying the observations has a fundamental advantage, especially in high dimensional situations in terms of computer memory allocation, as the full data need not be stored. In some high dimensional simulation experiments conducted in the paper, indeed L_1-SVM and SCAD SVM crashed due memory issues or numerical instabilities, but the proposed procedures always gave sensible output. More-

over, the proposed methods give significantly leaner models without compromising classification error. This is tremendous advantage because of better interpretability of simpler models and more stable parameter estimation.

In addition to the proximal SVM, the new algorithms can also be extended to other variations of the hinge loss, for example, the huberized hinge loss functions (Rosset and Zhu 2007; Wang et al. 2008). It will be also interesting to consider multiclass support vector machines (Lin et al. 2005) and grouped variable selection (Yuan and Lin 2006). Finally, it will be an attractive extension to develop a solution path for the proposed iterative methods.

7 Appendix

Proof of (7): Let $F(b) = N_+(1-b)_+ + N_-(1+b)_+$. Then F is continuous, convex, differentiable except at ± 1 and piecewise linear in $(-\infty,-1)$, $(-1,1)$ and $(1,\infty)$. Further under the conditions that $N_+, N_- > 0$, clearly $F(b) \to \infty$ as $b \to \pm\infty$ and $F(b) \geq F(-1)$ for any $b \leq -1$, $F(b) \geq F(1)$ for any $b \geq 1$. Thus F has a minimizer in $[-1,1]$. If $-1 \leq b \leq 1$, $F(b) = N_+ + N_- + b(N_- - N_+)$. If $N_+ = N_-$, then all $b \in [-1,1]$ are minima. If $N_- > N_+$, then F is minimized at $b = -1$, while if $N_- < N_+$, then F is minimized at $b = 1$. Putting all these together, the conclusion follows.

Proof of (9)–(11): Let $F(b) = \sum_{i:Y_i=1}(c_i - bX_{ij})_+ + \sum_{i:Y_i=-1}(c_i + bX_{ij})_+ + \lambda|b|$. Then it follows that F is continuous, convex, differentiable except at the knots $\{c_iY_iX_{ij}^{-1} : i = 1,\ldots,n\}$ and piecewise linear in between. Consequently, F' exists everywhere except at the knots, both sided derivatives exist at the knots, F' is piecewise constant and $F'(b-) \leq F'(b+)$ for any b. Then the minimizer is given by (9), and is also unique unless the solution is one of the two extreme knots.

To prove (10) and (11), further split the sum in $F(b)$ according to the cases $X_{ij} > 0$ and $X_{ij} < 0$. Note that the cases $X_{ij} = 0$ do not contribute to any variation due to changing b. Now differentiate $F(b)$ at any point other than the knots. Then $F'(b+)$ and $F'(b-)$ at any b including the knots can be obtained respectively from the right and left limiting procedures.

Proof of (14)–(15): Let T_1, T_2, T_3, T_4 denote the first four terms in the sum (11) without their signs, i.e., $F'(b-) = -T_1 - T_2 + T_3 + T_4 - \lambda\,\mathrm{sign}(b-)$. On the negative side, moving from a knot ξ to the adjacent one ξ^* on the left, T_1 gains terms corresponding to $Y_i = 1$, $X_{ij} > 0$, $c_iX_{ij}^{-1} = \xi^*$, T_2 loses terms corresponding to $Y_i = 1$, $X_{ij} < 0$, $c_iX_{ij}^{-1} = \xi^*$, T_3 loses terms corresponding to $Y_i = -1$, $X_{ij} > 0$, $c_iX_{ij}^{-1} = -\xi^*$, T_4 gains terms corresponding to $Y_i = -1$, $X_{ij} < 0$, $c_iX_{ij}^{-1} = -\xi^*$. Putting

these together, (14) follows.

The proof of (15) is similar. Let T_1, T_2, T_3, T_4 denote the first four terms in the sum (10) without their signs, i.e., $F'(b+) = -T_1 - T_2 + T_3 + T_4 + \lambda \,\mathrm{sign}(b-)$. On the negative side, moving from a knot ξ to the adjacent one ξ^* on the right, T_1 loses terms corresponding to $Y_i = 1$, $X_{ij} > 0$, $c_i X_{ij}^{-1} = \xi^*$, T_2 gains terms corresponding to $Y_i = 1$, $X_{ij} < 0$, $c_i X_{ij}^{-1} = \xi^*$, T_3 gains terms corresponding to $Y_i = -1$, $X_{ij} > 0$, $c_i X_{ij}^{-1} = -\xi^*$, T_4 loses terms corresponding to $Y_i = -1$, $X_{ij} < 0$, $c_i X_{ij}^{-1} = -\xi^*$. Putting these together, (15) follows.

Proof of (19): Let $F(b) = (1-b)_+^2 N_+ + (1+b)_+^2 N_-$. Clearly, $F(b) \geq F(-1)$ for all $b \leq -1$ and $F(b) \geq F(1)$ for all $b \geq 1$, so it suffices to study $F(b)$ in $[-1, 1]$, where

$$F(b) = N_+ + N_- + 2b(N_- - N_+) + b^2(N_+ + N_-) = N_+ + N_- + 2b(N_- - N_+) + b^2 n.$$

Thus the unique minima is given by $(N_+ - N_-)/n$.

References

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D. and Levine, A. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci. USA.*, **96**, 6745–6750.

Bühlmann, P. (2006). Boosting for high-dimensional linear models. *Ann. Statist.* **34**, 559–583.

Breiman, L. (1995). Better subset selection using the non-negative garotte. *Technometrics* **37**, 373–384.

Efron, B., Hastie, T., Johnstone, I. and Tibshirani, R. (2004). Least angle regression. *Ann. Statist.* **24**, 407–499.

Freund, Y., Schapire, R., and Abe, N. (1999). A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, **14**, 771-780.

Friedman, J., Hastie, T., Hofling, H. and Tibshirani, R. (2007). Pathwise coordinate optimization. *Ann. Appl. Statist.* **1**, 302–332.

Friedman, J., Hastie, T. and Tibshirani, R. (2010) Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software* **33**, 1-22.

Fung, G. and Mangasarian, O. (2001). Proximal support vector machine classifiers. *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 77 - 86.

Gao, Y., Fan, J., Xue, X., and Jain, R. (2006). Automatic image annotation by incorporating feature hierarchy and boosting to scale up SVM classifiers. In *Proceedings of the 14th annual ACM international conference on Multimedia*, 901-910, ACM.

Hwang, W-Y., Zhang, H. H. and Ghosal, S. (2009). FIRST: combining forward iterative selection and shrinkage in high dimensional sparse linear regression. *Statistics and Its Interface* **2**, 341–348.

Liu, Y., Shen, X., and Doss, H. (2005). Multicategory psi-learning and support vector machine: computational tools. *Journal of Computational and Graphical Statistics* **14**, 219-236.

Liu, Y. and Wu, Y. (2007). Variable selection via a combination of the L0 and L1 penalties. *Journal of Computational and Graphical Statistics* **16**, 782-798.

Mason, L., Baxter, J., Bartlett, P. and Frean, M. (1999). Functional Gradient Techniques for Combining Hypotheses. *In Advance in Large Margin Classifiers. MIT Press.*

Rosset,S. and Zhu,J. (2007) Piecewise linear regularized solution paths. *Annals of Statistics* **35**, 1012-1030.

Singh, D., Febbo, P., Ross, K., Jackson, D., Manola, J., Ladd, C., Tamayo, P., Renshaw, A., D'Amico, A., and Richie, J. (2002). Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell* **1**, 203-209.

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *J. Roy. Statist. Soc., Ser. B* **58**, 147-169.

Wang, H., Li, G., and Jiang, G. (2007). Robust regression shrinkage and consistent variable selection via the lad-lasso. *Journal of Business and Economic Statistics* **20**, 347355.

Wang, L., Zhu, J. and Zou, H. (2008). Hybrid huberized Support Vector Machines for microarray classification and gene selection. *Bioinformatics* **24**, 412-419.

Yuan, M. and Lin, Y. (2006) Model selection and estimation in regression with grouped variables. *Journal of Royal Statistical Society, B.* **68**, 4967.

Zhang, H. H., Ahn, J., Lin, X., and Park, C. (2006). Gene selection using Support Vector Machines with nonconvex penalty. *Bioinformatics* **22**, 88-95.

Zhu, J., Rosset, S., Hastie, T. and Tibshirani, R. (2004). 1-norm support vector machines. *The Annual Conference on Neural Information Processing Systems* **16**.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of Royal Statistical Society, Series. B*, **67**, 301–320.

Zou H. (2006). The adaptive Lasso and its oracle properties. *Journal of American Statistical Association* **476**, 1418–1429.

Zou, H. (2007). An improved 1-norm Support Vector Machine for simultaneous classification and variable selection. *Journal of Machine Learning Research – Proceedings Track 2*, 675–681.