ABSTRACT

LIU, SONG. Variable Selection in Semi-parametric Additive Models with Extensions to High Dimensional Data and Additive Cox Models . (Under the direction of Dr. Hao Helen Zhang).

Variable selection in nonparametric and semi-parametric regression is more challenging than in linear models or other parametric models. We not only need to estimate the component functions with proper smoothers, but also consider which terms should be included in the model. In this paper, we focus on the variable selection in semi-parametric additive models. We propose new methods of variable selection for generalized additive models (GAM) and additive Cox models based on Breiman's (1995) nonnegative garrote. To achieve optimal performance of garrote in GAMs and the additive Cox models, we develop several algorithms to improve the initial estimation of component functions. Our methods can deal with continuous and categorical variables in a unified fashion.

High dimensional data are more and more encountered in real life, and posing serious challenge for model estimation and variable selection. In this dissertation research, we extend our methods to high dimensional data by developing a two-stage garrote to conduct variable selection and estimation. Simulations and real examples show that our methods are very competitive in terms of prediction and variable selection compared with other variable selection methods in semi-parametric additive models.

Variable Selection in Semi-parametric Additive Models with Extensions to High Dimensional Data and Additive Cox Models

by
Song Liu

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fullfillment of the
requirements for the Degree of
Doctor of Philosophy

Statistics

Raleigh, North Carolina

2008

APPROVED BY:

Wenbin Lu	John Monahan
Hao Helen Zhang Chair of Advisory Committee	Dennis Boos

DEDICATION

To Yun, Elvin and my parents

BIOGRAPHY

Song Liu was born in Handan, Hebei, China on October 29, 1975. He attended Nankai University in China in 1994. He earned a Bachelor of Science degree in Biology in 1998, and Master degree of Genetics in 2001. He then went to Penn State University to study Genetics and Applied Statistics, until 2004. Thereafter, he came to North Carolina State University and received a Master of Science degree in statistics in May, 2006. He continued his studies towards a Ph.D degree in statistics at North Carolina State University. After he finishes his Ph.D study, he will join Travelers at Hartford, CT, in May 2008.

ACKNOWLEDGMENTS

I express sincere gratitude to my advisor Dr. Hao Helen Zhang for introducing me to the broad spectrum of statistics, including nonparametric regression and data mining research. It has been a great pleasure to have discussion with her for years. This has widened my knowledge and enlightened my interest in variable selection. I express sincere appreciation to Dr. Boos, Dr. Wenbin Lu and Dr. John Monahan for being my Ph.D. committee members, reviewing my dissertation, providing helpful comments and great teaching during my graduate career.

Thanks to Dr. Pam Arroway, Dr. Jacqueline Hughes-Oliver and Adrian Blue, for their support and assistance of my graduate study.

Special thank goes to my wife Yun Wu. Without her unreserved support and love, my Ph.D study can never be finished. I would also like to thank my parents, they have always supported and believed in me and guided me. Last but not least I would like to thank my lovely son, Elvin. He brings endless joy and happiness into my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Variable selection is an important topic in statistical modeling. It is an effective way to reduce model complexity by balancing model bias and model variance. Identifying the most important predictors will improve both prediction accuracy and model interpretability. For linear models or parametric models, statisticians often use classical methods to choose important variables such as forward selection, backward elimination and best-subset selection. But these procedures are known to be locally optimal and perform with high variability (Breiman, 1995).

In recent years, a lot of regularization techniques are applied to variable selection, and many new methods are developed (Breiman, 1995; Tibshirani, 1996; Fan and Li, 2001; Efron et al., 2004; Yuan and Lin, 2006; Zou, 2006). Particularly, these methods constrain the "length" of the coefficients of predictors to be some preselected values which are tuning parameters. By properly choosing tuning parameters, we can find a sparse model with improvement in the prediction errors (PE) over the full model. We define prediction error in the same way as in Tibshirani (1996). It is assumed that the observation $(\mathbf{X}, Y)$ are drawn from some unknown distribution, where $\mathbf{X}$ is a p-dimensional predictor, Y is a response. For convenience, we assume $\mathbf{X}$ is fixed. Suppose that

$$Y = \eta(\mathbf{X}) + \epsilon, \tag{1.1}$$

where $E(\epsilon) = 0$ and $Var(\epsilon) = \sigma^2$. Then the PE of an estimate $\hat{\eta}(\mathbf{X})$ is defined as

$$PE = E\{Y - \hat{\eta}(\mathbf{X})\}^2 = ME + \sigma^2, \tag{1.2}$$

the expectation is taken over the conditional distribution of Y given $\mathbf{X}$, if $\mathbf{X}$ is fixed. The model error (ME) is given by

$$ME = E\{\hat{\eta}(\mathbf{X}) - \eta(\mathbf{X})\}^2. \tag{1.3}$$

This component is the prediction error due to lack of fit to the underlying true model. We usually use the estimated ME to evaluate models.

There are several approaches to estimate ME. Among these methods, Mallows C_p (Mallows, 1973), Akaike Information Criteria (Akaike, 1970, 1974) and Bayesian Information Criteria (Schwarz, 1978) are most commonly used. Other popular methods include Cross Validation (Picard and Cook, 1984) and Bootstrap (Efron, 1977), and both of these two methods are computationally intensive.

Consider a simple linear regression problem with n observations $\{\mathbf{x}_i, y_i\}_{i=1}^n$, where $\mathbf{x}_i = (x_{1i}, \ldots, x_{pi})$ are linearly independent predictors, and

$$y_i = \sum_{j=1}^{p} x_{ji}\beta_j + \varepsilon_i, \qquad i = 1, 2, \ldots, n, \tag{1.4}$$

where ε_i's are iid random errors. For convenience, we do not include the intercept in the model and assume the variable y and x_j's, $j = 1, \ldots, p$, are all centered.

Fan and Li (2001) discussed the desired properties of a good variable selection procedure, including the oracle properties. Assume the underlying true model for (1.4) has a sparse representation. Without loss of generality, let $\boldsymbol{\pi} = \{j : \beta_j \neq 0\} = \{1, 2, \ldots, p_0\}$ and $p_0 < p$. We also define $\boldsymbol{\pi}^c = \{j : \beta_j = 0\}$ as the true index set of zero parameters. Therefore, the true parameter $\boldsymbol{\beta}$ has a partition $(\boldsymbol{\beta}_\pi, 0)$, and correspondingly, we decompose any estimator $\widehat{\boldsymbol{\beta}} = (\widehat{\boldsymbol{\beta}}_\pi, \widehat{\boldsymbol{\beta}}_{\pi^c})$. Define $\boldsymbol{\pi}^* = \{j : \widehat{\boldsymbol{\beta}}_j \neq 0\}$. Under some mild regularity conditions in Appendix of Fan and Li (2001), we call $\widehat{\boldsymbol{\beta}}$ an oracle estimator if it satisfies the following conditions:

1. Consistence in variable selection: $\lim_{n\to\infty} P(\boldsymbol{\pi}^* = \boldsymbol{\pi}) = 1$;

2. Asymptotic normality: $\sqrt{n}\left(\widehat{\boldsymbol{\beta}}_{\boldsymbol{\pi}} - \boldsymbol{\beta}_{\boldsymbol{\pi}}\right) \rightarrow_d N(0, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is the $p_0 \times p_0$ covariance matrix of the estimator for the nonzero coefficients if the true subset mode were known.

The procedure with oracle properties is considered consistent in terms of both model estimation and variable selection.

1.1 Penalized Least Squares for Linear Models

Breiman (1995) introduced the nonnegative garrote (nn-garrote) estimator in linear models. We denote the ordinary least squares (OLS) estimator of $\boldsymbol{\beta}$ as $\widehat{\boldsymbol{\beta}}^{\text{OLS}}$. The original nn-garrote estimator is a "shrunk" OLS estimator. The shrinking factor c_j's are the solution of

$$\min_{c_1,\ldots,c_p} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} c_j x_{ji} \hat{\beta}_j^{\text{OLS}} \right)^2 + \lambda \sum_{j=1}^{p} c_j, \qquad \text{subject to } c_j \geq 0, \text{ for all } j, \quad (1.5)$$

where $\lambda > 0$ is a tuning parameters. The nn-garrote estimator is defined as $\hat{\beta}_j^g = \hat{c}_j \hat{\beta}_j^{\text{OLS}}$, $j = 1, \ldots, p$. When we have the orthogonal design, the garrote shrinking factor has an explicit form:

$$\hat{c}_j = \left(1 - \frac{\lambda}{\left(\hat{\beta}_j^{\text{OLS}}\right)^2} \right)_+ . \quad (1.6)$$

From (1.6), the coefficients with large OLS estimates will have large $\hat{c}_j$ close to 1, while the coefficients with small OLS estimates tend to have small $\hat{c}_j$, which can be exactly 0 when $|\hat{\beta}_j^{\text{OLS}}| \leq \sqrt{\lambda}$. Therefore, the garrote estimator has a sparse solution. Besides this sparsity property, Breiman (1995) conducted comprehensive simulations to compare subset selection and the nn-garrote estimator. Both simulated and real experiments showed that the nn-garrote estimator has the mean prediction error smaller than that of the subset selection, and comparable to that of the ridge regression. Breiman (1995) concluded that subset selection is very unstable in terms of variable selection, while the nn-garrote estimator is much more stable.

The garrote has a drawback, as pointed by Tibshirani (1996), that it heavily depends on the OLS estimates. When the OLS estimators behave badly in some situations such as highly correlated covariates, the garrote estimates may suffer the same problems as the OLS estimates. Tibshirani (1996) introduced the Least Absolute Selection and Smoothing Operator (LASSO) estimate, which does not depend on the OLS estimates. The Lasso estimate is defined as

$$\widehat{\boldsymbol{\beta}}^{\text{lasso}} = \min_{\beta_1,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ji}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \,, \tag{1.7}$$

where $\lambda \geq 0$ is a tuning parameter. The second term in (1.7) is called the ℓ_1 penalty. With the ℓ_1 penalty, the lasso can do variable selection and estimation simultaneously and it has brought a lot of interest in the statistical machine learning community.

Many other techniques based on the ℓ_1 penalty have been developed recently (Fu, 1998; Fan and Li, 2001; Efron *et al.*, 2004; Yuan and Lin, 2006; Zou and Hastie, 2005; Zou, 2006), and theoretical results on the consistency properties of the lasso estimate are established. Fan and Li (2001) conjectured that the lasso estimate does not have oracle properties and Zou (2006) proved that the lasso is not an oracle estimate in general although the lasso estimate is consistent in terms of estimation.

Fan and Li (2001) proposed a new variable selection approach to remedy the inconsistency of the lasso via a nonconcave penalized likelihood. They imposed a new penalty, which is derived from ℓ_1 penalty, named Smoothly Clipped Absolute Deviation (SCAD), on the log likelihood function. With the SCAD penalty, the coefficients of unimportant variables are shrunk to be exactly 0 and the important coefficients are not severely biased as the lasso estimates. Fan and Li (2001) proved the SCAD estimate is consistent for both estimation and variable selection under certain regularity conditions. So the SCAD estimate has oracle properties.

Zou (2006) proposed the adaptive lasso to fix the consistency problem of the original lasso. The adaptive lasso is defined as:

$$\widehat{\boldsymbol{\beta}}^{\text{alasso}} = \min_{\beta_1,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ji}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \frac{|\beta_j|}{|\tilde{\beta}_j|}, \tag{1.8}$$

where $\tilde{\beta}_j$ is a $\sqrt{n}$-consistent estimator of β_j, $j = 1, \ldots, p$. Zou (2006) showed that the adaptive lasso enjoyed the oracle properties, and further connected his adaptive lasso with the nonnegative garrote.

In (1.8), if we replace $\tilde{\beta}_j$ with the OLS estimators $\hat{\beta}_j^{\text{OLS}}$, then the adaptive lasso solves

$$\min_{\beta_1,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ji}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \frac{|\beta_j|}{|\hat{\beta}_j^{\text{OLS}}|}. \tag{1.9}$$

This equation is similar to (1.5). Since $\widehat{c_j} = \hat{\beta}_j^g / \hat{\beta}_j^{\text{OLS}}$, we can reformulate (1.5) as

$$\min_{\beta_1,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ji}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \frac{|\beta_j|}{|\hat{\beta}_j^{\text{OLS}}|}, \text{ subject to } \beta_j \hat{\beta}_j^{\text{OLS}} \geq 0, \forall j. \tag{1.10}$$

The equations (1.9) and (1.10) are almost the same except the sign constraint in (1.10). Zou (2006) proved the garrote estimate is consistent both in estimation and in variable selection, and Yuan and Lin (2007) conducted an independent study on its theoretical and computational properties in great details.

Yuan and Lin (2007) extended the garrote into a more general case:

$$\min_{c_1,\ldots,c_p} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} c_j x_{ji} \hat{\beta}_j^{\text{init}} \right)^2 + \lambda \sum_{j=1}^{p} c_j, \text{ subject to } c_j \geq 0, j = 1, \ \ldots, p, \tag{1.11}$$

where $\hat{\beta}_j^{\text{init}}$ is an initial estimate. This initial estimate is not restricted to the least squares estimate, and other estimates including lasso, ridge regression and elastic net (Zou and Hastie, 2005) can also be used as initial values. They proved that the solution to (1.11) has oracle properties as long as the initial estimate is $\sqrt{n}$-consistent in estimation and suggested a path algorithm which is similar to LARS solution (Efron *et al*, 2004).

Yuan and Lin (2006) proposed the group garrote to select groups of variables (factors) in linear models. The idea is to put garrote-type constraints groupwisely instead of elementwisely on initial estimates. Therefore, variables in a same group can be either selected or discarded as a bundle. A detailed review on the group garrote is given in Chapter 2.

1.2 Variable Selection for Nonparametric Models

Currently, most methods of variable selection focus on linear models or parametric models. Nonparametric and semi-parametric regression models allow more relaxed assumptions than linear models, but not many methods have been developed for variable selection in nonparametric and semi-parameter regression models.

The streamlined techniques include backward and forward stepwise selection, which are implemented in CART (Breiman et al., 1984), TURBO (Friedman and Silverman, 1989), BRUTO (Hastie,1989), and MARS (Friedman,1991). These methods perform well in practice, but they can be computationally expensive because they all use some types of search algorithms. In addition, as the forward selection and backward selection in linear models, they are discrete processes that either keep or drop one variable at a time. In this way, they may have problems in finding the globally optimal solution.

Kohn and his colleagues (Shively, Kohn and Wood,1999; Wood, Kohn, Shively and Jiang, 2002; Yau, Kohn and Wood, 2003; Yaui and Kohn, 2003) proposed a MCMC based Bayesian method to conduct variable selection for additive models. Their experiment results are very promising, but their approach is computationally demanding. Zhang *et al.* (2004) introduced the likelihood basis pursuit approach for model selection and estimation in the functional ANOVA. Their method is also computationally expensive.

The regularization techniques are continuous in variable selection, and usually result in optimal models with low variability and good prediction. Compared to traditional search algorithms, the success of regularization methods in linear models provides new motivations for variable selection in nonparametric models. In the framework of smoothing spline ANOVA (SS-ANOVA), Lin and Zhang (2006) developed a lasso-type method—COSSO, Component Selection and Smoothing Operator, to conduct variable selection and function estimation simultaneously. The COSSO allows the treatment of Gaussian data, exponential families (Zhang and Lin, 2006) and survival data (Leng and Zhang, 2006). In the SS-ANOVA COSSO model, the penalty is the sum of functional component norms and referred to as the COSSO

penalty. Like the lasso penalty, the COSSO penalty leads to a sparse model. Since COSSO is based on smoothing splines, the original full basis algorithm for COSSO generally demands high computational cost when the sample size is large. To address this issue, Zhang and Lin (2006) suggested the subset basis algorithm, which effectively reduces the computation burden of the COSSO.

Compared with the lasso penalty, the garrote penalty has many advantages if a good initial estimate is available. Firstly, the garrote estimate is shown to have oracle properties for linear models. Secondly, it is easy and natural to extend to any dictionary-based models, for example, additive models in both nonparametric and semi-parametric settings. Cantoni *et al.* (2006) adapted the garrote method to nonparametric additive models with Gaussian data. They used several fitting methods to get initial estimates and 5-fold cross validation to find tuning parameters in the garrote. Their simulation results showed that the garrote estimate in additive models gives competitive performance in terms of prediction error compared to the COSSO, but tends to overfit in terms of variable selection. However, Cantoni et al. (2006) can not handle non-normal data, like count data, binary data and survival data with censoring.

Recently, Yuan (2008) also extended the nn-garrote for component selection in additive models and more general functional ANOVA models. He used smoothing splines to get initial estimates, and developed a similar path algorithm as LARS (Efron *et al.*, 2004) to get the whole path solution efficiently. Most importantly, he showed that the garrote estimate enjoys the good oracle properties as in the linear models given the tuning parameter is appropriately chosen. In Theorem 2 of Yuan (2008), Yuan pointed out that as long as the initial estimate of each component is consistent, the garrote estimate is consistent in terms of both estimation and variable selection. In other words, the garrote has the ability to "transform" a consistent initial estimate to the estimate with oracle properties. Again, Yuan's work focused on normal data only.

1.3 Proposal of Work

In this dissertation study, we develop new variable selection methods in general semi-parametric settings of exponential family data and survival data. Furthermore, we extend our methods to high dimensional data. Model estimation and variable selection for these types of data pose several unique challenges:

- For non-normal data, statistical inferences are often based on likelihood functions. The garrote in the context of penalized least squares then needs to be extended to the general case of penalized likelihood. Both inferences and computation are more complicated than the least squares setting.

- Not many methods are available for estimation and variable selection for the nonparametric and semi-parametric Cox proportional hazard models.

- When the sample size is small and there are many noisy variables, the current algorithm for fitting generalized additive models often encounters convergence problems, which makes the estimation results very unstable. A more robust method for small data is therefore desired.

In our work, we will address all the issues above and extend the garrote method to generalized additive models (GAM), to survival data and high dimensional data. We will propose new techniques to improve both initial estimates and the method of tuning. In addition, we will use the group garrote to deal with categorical variables altogether with continuous ones. Efficient algorithms are developed for computation. We suggest a new boosting algorithm to obtain a stable initial GAM estimates when there are a large number of covariates. In addition, we also propose a two-stage garrote method for high dimensional data. Our simulation results and real examples show that the new methods perform as well or better than other competitors in terms of both prediction and variable selection.

This thesis is organized as follows. In Chapter 2, our new methods of variable selection in semi-parametric GAMs are introduced. We first review some background material for GAMs, the group garrote and boosting, then propose our methods and

algorithms, followed by simulations and real examples. In Chapter 3, we propose the garrote method for additive Cox models. In Chapter 4, we propose a new variable selection for high dimensional data. Chapter 5 contains conclusions and comments.

Chapter 2

New Variable Selection for Semi-parametric Additive Models

In this chapter, we propose a new variable selection method for semi-parametric regression models in the setting of penalized likelihood. We also develop a new algorithm to get the stable initial estimates for GAMs. The computation and tuning parameter issues are discussed. Simulation and real examples are conducted.

2.1 Review on GAMs

Generalized additive models (Hastie and Tibshirani, 1990) are a class of extended generalized linear models with a linear predictor replaced by a sum of unknown smooth functions of predictors. Suppose we have a response variable Y and p-dimensional covariate $\mathbf{X} = (X_1, \ldots, X_p)$. Assume $Y|\mathbf{X}$ has an exponential family distribution of

$$\exp\left(y\eta(\mathbf{X}) - B(\mathbf{X}) + C(y)\right), \tag{2.1}$$

where B and C are known functions. The purpose of regression is to estimate the unknown $\eta(\mathbf{X})$ based on an independently and identically distributed sample. The GAM assumes the structure of:

$$g(\mu) = \eta(\mathbf{X}) = \alpha + \sum_{j=1}^{p} f_j(X_j), \tag{2.2}$$

where $\mu = E(Y|\mathbf{X})$, $g(\cdot)$ is a known link function, α is the intercept, and f_j, $j = 1, \ldots, p$, are unspecified univariate functions. This model allows a more relaxed assumption on the function form of covariates than parametric models, in which we assume the response has a parametric relationship with predictors. This relaxed assumption brings two important issues for GAMs: fitting the GAMs with proper smoothers and choices of smoothing parameters. In the following, we review several popular approaches to the model estimation for GAMs.

2.1.1 GAMs with Backfitting

Hastie and Tibshirani (1990) first proposed solving GAMs with the so-called backfitting technique. This simple method has the advantage of allowing one to estimate the component functions with any smoothing technique. Now we give a brief description of this algorithm. We first illustrate how the backfitting algorithm works in additive models. This is Gauss-Seidel type of updating procedure (Monahan, 2001). Assume we have data $\{y_i, \mathbf{x}_i\}_{i=1}^n$ following the model,

$$y_i = \alpha + \sum_{j=1}^{p} f(x_{ji}) + \varepsilon_i, \ i = 1, \ldots, n, \tag{2.3}$$

where $\varepsilon_i, \ i = 1, \ldots, n$, are iid errors.

Define $\mathbf{y} = (y_1, \ldots, y_n)^T$ and $\mathbf{f}_j = (f_j(x_{j1}), \ldots, f_j(x_{jn}))^T$ for $j = 1, \ldots, p$. The backfitting algorithm is:

1. For $j = 1, \ldots, p$, set $\hat{\alpha} = \overline{y}$ and $\hat{\mathbf{f}}_j^{(1)} = \mathbf{0}$.

2. For $m = 1, 2, \ldots$, until all $\hat{\mathbf{f}}_j$ converge

   For $j = 1, 2, \ldots, p$

$$\hat{\mathbf{f}}_{j,\text{temp}} = \mathbf{S}_j[\mathbf{y} - \hat{\alpha} \cdot \mathbf{1} - \sum_{k<j} \hat{\mathbf{f}}_k^{(m+1)} - \sum_{k>j} \hat{\mathbf{f}}_k^{(m)}], \tag{2.4}$$

$$\hat{\mathbf{f}}_j^{(m+1)} = \hat{\mathbf{f}}_{j,\text{temp}} - \frac{1}{n} \sum_{i=1}^{n} \hat{f}_{j,\text{temp}}(x_{ij}) \cdot \mathbf{1}, \tag{2.5}$$

   end (inner For loop);

end (outer For loop).

where $\mathbf{S}_j$ is the smoother for the jth variable. To fit the GAM, a weighted version of backfitting is applied. The algorithm is known as the iteratively reweighted least squares (IRLS) (Hastie and Tibshirani, 1990).

There is an R package *gam* providing backfitting GAMs. Although the backfitting algorithm chooses a wide range of smoothers, the tuning can be computationally expensive. There are multiple tuning parameters, one associated with each component, and it is often hard to determine the proper amount of smoothing for individual component functions. Commonly used tuning criteria include Cross Validation (CV) and Generalized Cross Validation (GCV).

2.1.2 GAMs with Penalized Regression Splines

To facilitate the problem of choosing smoothing parameters in backfitting, Wood (2000, 2004) developed another method based on penalized regression splines to fit GAMs while incorporating automatic selection of smoothing parameters at the same time. In this method, the smoothing functions are restricted to regression splines, which requires the specification of the number of knots and their locations.

Wood (2003) proposed a new smoothing method, thin plate regression spline, which is a hybrid of thin-plate smoothing splines and regression splines. The thin-plate regression spline avoids the problem of placing knots and provides low rank approximations to generalized smoothing spline models. Wood (2003) argued that the performance of the thin-plate regression splines is not sensitive to the dimension of basis. The advantage of this setting is that we can automatically choose smoothing parameters for each smoother with low computational cost. More details are given as follows.

- First, we represent each smooth term with a basis expansion,

$$f_j(x_{ji}) = \sum_{k=1}^{K_j} r_{jk} b_{jk}(x_{ji}), i = 1, \ldots, n, j = 1, \ldots, p, \tag{2.6}$$

where b_{jk} is a preselected basis function, K_j is the number of the basis functions to estimate f_j. Absorb $b_{jk}(x_{ji})$ into the design matrix $\mathbf{B}$, and r_{jk} into $\boldsymbol{r}$, $j = 1, \ldots, p$, $i = 1, \ldots, n$, $k = 1, \ldots, K_j$, then the model in (2.3) has the form of a linear model:

$$\mathbf{y} = \mathbf{B}\boldsymbol{r} + \boldsymbol{\varepsilon}, \tag{2.7}$$

where $\boldsymbol{\varepsilon}$ is an n-vector of iid random errors.

- As in the setting of smoothing splines, we estimate $\boldsymbol{r}$ by minimizing:

$$\|\mathbf{y} - \mathbf{B}\boldsymbol{r}\|^2 + \sum_{j=1}^{p} \lambda_j \int \ddot{f}_j^{\,2}(x)\mathrm{d}x, \tag{2.8}$$

  where λ_j is a smoothing parameter for the jth term, $j = 1, \ldots, p$. The second term of (2.8) can be rewritten as $\boldsymbol{r}^{\mathrm{T}}\boldsymbol{\Omega}_j\boldsymbol{r}$ where $\boldsymbol{\Omega}_j$ can be expressed in terms of the basis functions $\mathbf{b}_j = (b_{j1}, \ldots, b_{jK_j})$, $j = 1, \ldots, p$. To avoid the identifiability problem, we may impose the mean-zero constraints on f_j, $j = 1, \ldots, p$. Then we estimate $\boldsymbol{r}$ by minimizing

$$\|\mathbf{y} - \mathbf{B}\boldsymbol{r}\|^2 + \sum_{j=1}^{p} \lambda_j \boldsymbol{r}^{\mathrm{T}}\boldsymbol{\Omega}_j\boldsymbol{r},$$

  which is a generalized ridge regression for a given set of λ_j, $j = 1, \ldots, p$.

- Choose the multiple smoothing parameters by minimizing the GCV score with the modified Newton method, and estimate the component functions with selected smoothing parameters.

To fit a GAM, we use the weighted version of the above procedure. The algorithm of estimating multiple smoothing parameters is fast, taking $p^3 \sum_{j=1}^{p} K_j^3$ operations, where p is the number of smoothing terms and K_j is the dimension of basis function for each term. In our method of variable selection, we use this GAM method to get initial estimates. In R, the package *mgcv* provides the function *gam* to implement this algorithm.

In addition to these two methods of fitting GAMs, there are other alternatives. The generalized smoothing spline method of Wahba (1990) and Gu (2002) is built

on a complete theoretical framework. Their approach is based on the elegant theory of reproducing kernel Hilbert space (RKHS). The drawback of their approach is high computational cost. Gu (2002) developed the *gss* package in R to fit the GAM. Fahrmeir and Lang (2001) and Fahrmeir et al. (2004) suggested a Bayesian approach for GAMs and used MCMC technique for computation.

2.2 Other Background Review

2.2.1 Group Garrote for Linear Models

In linear models, multifactor or categorical variables are often coded as a group of dummy variables. Traditional variable selection methods usually select the dummy variables separately without taking into account their internal relationship. Yuan and Lin (2006) proposed the group selection idea by encouraging the selection of dummy variables associated with one same variable simultaneously. The following gives a brief introduction on the group garrote.

Consider a regression model with J factors,

$$\mathbf{y} = \sum_{j=1}^{J} X_j \boldsymbol{\beta}_j + \varepsilon, \tag{2.9}$$

where $\mathbf{y}$ is a length of n vector, $\varepsilon \sim N(0, \sigma^2 I)$, X_j is an $n \times p_j$ matrix corresponding to the jth factor, and $\boldsymbol{\beta}_j$ is a coefficient vector of size p_j, $j = 1, \ldots, J$. Let $\widehat{\boldsymbol{\beta}}_j^{\text{OLS}}$ denote the OLS estimate of $\boldsymbol{\beta}_j$, and $Z_j = X_j \widehat{\boldsymbol{\beta}}_j^{\text{OLS}}$, $Z = (Z_1, \ldots, Z_J)$. Then the group garrote estimate $\hat{\mathbf{d}} = (\hat{d}_1, \ldots, \hat{d}_J)^T$ is the solution of

$$\min_{\mathbf{d}} \left(\|y - Zd\|^2 + \lambda \sum_{j=1}^{J} p_j d_j \right), \text{ subject to } d_j \geq 0, \forall j. \tag{2.10}$$

The final prediction is given by $\hat{\mathbf{y}} = \sum_{j=1}^{J} X_j \widehat{\boldsymbol{\beta}}_j^{\text{OLS}} \widehat{d}_j$. If $\widehat{d}_j = 0$, then the entire group of dummy variables associated with X_j is dropped out of the final model. Obviously, when $p_1 = \cdots = p_J = 1$, this model reduces to the ordinary garrote proposed by Breiman (1995). The group garrote solution can be solved by quadratic programming.

Yuan and Lin (2007) developed a path algorithm for the garrote, which can be also applied in the group garrote. To choose the tuning parameter λ, Yuan and Lin (2006) proposed an approximation formula to calculate the degree of freedom (df) for the group garrote,

$$\tilde{\text{df}} = 2\sum_{j=1}^{J} I(\hat{d}_j > 0) + \sum_{j=1}^{J} \hat{d}_j(p_j - 2), \tag{2.11}$$

where I is an indicator function.

Yuan and Lin (2006) conducted simulations to compare the group garrote, group lasso and group LARS with stepwise selection, and concluded that the group garrote has smaller prediction errors and gives a more sparse model than other methods. The initial estimates in the group garrote can be replaced by other consistent estimates.

2.2.2 Boosting

Boosting is one of the most successful learning algorithms in statistical learning. It was first introduced by Scapire (1990) and Freund (1995) in the context of classification problems. The essential idea of the boosting method is to build a powerful "committee" by series of "weak" learners. Here the "weak" learner refers to a very simple classifier which is just slightly better than a random guess. Usually, when a model has small variance and large bias, the corresponding learner is considered to be "weak". Friedman et al. (2000) gave a detailed review on boosting from a statistical point of view, where the boosting is viewed as a functional gradient descent algorithm. The idea of boosting has its root in PAC (Probably Approximately Correct) learning (Valient, 1984). The *AdaBoost* (Adaptive Boosting) algorithm proposed by Freund and Schapire (1996) is generally considered as a first step towards the practical boosting algorithm. There are many types of boosting algorithms such as *gradientBoost* (Breiman, 1999; Friedman, 2001) and L_2 boosting (Buhlmann and Yu, 2005).

AdaBoost

Consider a classification problem with two classes, with the label $Y \in \{-1, 1\}$. A classifier $G_m(\mathbf{X})$ is obtained by training on the data with $p-$dimensional predictor

$\mathbf{X}$. Define $F(\mathbf{X}) = \sum_{m=1}^{M} \alpha_m G_m(\mathbf{X})$, in which m is a boosting iteration and α_m is the weight for each boosting iteration. After M steps, the boosting estimator is $sign(F(\mathbf{X}))$. In general, the *AdaBoost* will train a classifier based on different weights on the training sample, and give large weights to the sample points that are misclassified in the previous boosting iteration.

Freund and Schapire (1996) explained the *AdaBoost* with following algorithm.

1. Set initial weights $w_i = 1/n, i = 1, \ldots, n$.
2. For $m = 1, \ldots, M$:
   - Train $G_m(\mathbf{X})$ with weights w_i's on the data.
   - Calculate $e_m = (\sum_{i=1}^{n} w_i I(y_i \neq G_m(\mathbf{x}_i)))/(\sum_{i=1}^{n} w_i)$, and $\alpha_m = \log((1 - e_m)/e_m)$.
   - Recalculate weights: $w_i \leftarrow w_i \exp(\alpha_m I(y_i \neq G_m(\mathbf{x}_i))), i = 1, \ldots, n$, and do normalization so that $\sum_i w_i = 1$.
3. Get the classifier by calculating $sign(\sum_{m=1}^{M} \alpha_m G_m)$.

The total number of iterations M is usually decided by cross validation.

gradientBoost

Breiman (1999) first proposed a gradient decent boosting. Friedman (2001) developed a more general gradient boosting algorithm based on the weak learner tree, which is known as Multiple Additive Regression Tree (MART). Friedman (2001) further explained boosting as a numerical function optimization algorithm in certain functional spaces. He also connected the stagewise additive expansion and steepest descent.

Suppose our objective is to minimize a differentiable loss function $L(f)$. Then the steepest decent is $h_m = -\rho_m g_m$, where ρ_m is a scalar and g_m is the gradient of $L(f)$ evaluated at $f = f_{m-1}$. m is the boosting iteration. The ith component of g_m is

$$g_{mi} = -\left(\frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right)_{f(\mathbf{x}_i)=f_{m-1}(\mathbf{x}_i)}. \tag{2.12}$$

Then the step length ρ_m is the minimizer of $L(f_{m-1} + \rho g_m)$. The current solution is updated by

$$f_m = f_{m-1} + \rho_m g_m. \tag{2.13}$$

The detailed algorithm of generic *gradientBoost* is as following:

1. Initial $f_0(\mathbf{x}) = \min_\rho \sum_{i=1}^n L(y_i, \rho)$.

2. For $m = 1, \ldots, M$:

   - $\tilde{y}_i = -\left(\frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right)_{f(\mathbf{x}_i)=f_{m-1}(\mathbf{x}_i)}, i = 1, \ldots, n.$
   - $\gamma_m = \min\limits_{\gamma,\ \beta} \sum_{i=1}^n (\tilde{y}_i - \beta b(\mathbf{x}_i; \gamma))^2$.
   - $\rho_m = \min\limits_{\rho} {}^n_{i=1} L(y_i, f_{m-1}(\mathbf{x}_i) + \rho b(\mathbf{x}_i; \gamma_m))$.
   - $f_m(x) = f_{m-1}(\mathbf{x}) + \rho_m b(\mathbf{x}; \gamma_m)$.

3. Stop at $(M + 1)$th step .

Here $b(\mathbf{x}_i; .)$ is a basis function in regression tree.

Friedman (2001) conducted comprehensive experiments and showed that MART is a competitive and highly robust procedure for both regression and classification problems, and it outperforms many other data mining tools in cases of messy data.

L_2 Boosting

Buhlmann and Yu (2003, 2005) proposed L_2 boosting with componentwise least square and cubic smoothing splines as base learners.

In L_2 boosting, the loss function is the squared error loss $L(y, f) = (y - f)^2$. In this section, we choose the componentwise linear least squares as the base learners. Based on the data $\{y_i, \mathbf{x}_i\}$, $i = 1, \ldots, n$, the L_2 boosting algorithm conducts the following procedure:

1. Set $\hat{f}^{(0)} = \bar{y}$.

2. For $m = 1, \ldots, M$:

   - Compute the residuals $r_i = y_i - \hat{f}^{(m-1)}(\mathbf{x}_i)$, $i = 1, \ldots, n$.

- Fit the component OLS for each $\mathbf{x}_j$, and get the fitted coefficients $\hat{\beta}_j$, $\hat{\alpha}_j$, $j = 1, \ldots, p$.

  Select the covariate $\mathbf{x}_\kappa$ which reduces the L_2 loss most:

$$\kappa = \min_{j=1,\ldots,p} \|\mathbf{r} - \hat{\alpha}_j \mathbf{1} - \hat{\beta}_j \mathbf{x}_j\|^2,$$

  where $\mathbf{r} = (r_1, \ldots, r_n)^T$.

- Update $\hat{f}^{(m)}(\mathbf{x}) = \hat{f}^{(m-1)}(\mathbf{x}) + \nu(\hat{\alpha}_\kappa + \hat{\beta}_\kappa \mathbf{x}_\kappa)$.

The ν in the last step is called the shrinkage factor. The natural value of ν is 1, but smaller values are often preferred since they lead to more stable and less greedy procedures. Usually, the smaller we choose ν, the more boosting iterations we must take. Empirical results (Buhlmann and Yu, 2003, 2005) suggested that $\nu = 0.1$ is a good choice in practice.

The L_2 boosting with componentwise least squares works in the similar way as the lasso. It can conduct variable selection and component estimation simultaneously. The only tuning parameter is the boosting step. Buhlmann and Yu (2003) showed that L_2 boosting with componentwise smoothing splines can fit additive models with variable selection. They also proved that the boosting estimation of component function can achieve optimal convergence rate in one dimensional case, and can be adapted to higher-order smoothness.

Connection of Boosting and ℓ_1 Regularization

Recent boosting research showed there is a strong connection between boosting and ℓ_1 regularization (lasso). Zhang and Yu (2003) explored the quantitative relationship between early stopping in boosting and the ℓ_1 penalty. Zhang (2003) proposed a "shrinkage" boosting which converges to ℓ_1 solution path. Efron *et al.* (2004) showed that forward stagewise regression, which is similar to gradient boosting, is closely related to the lasso.

2.3 New Methodology

We consider the generalized additive model with both continuous and categorical variables. Since the statistical inferences of GAMs are generally based on the likelihood functions, we will use the penalized likelihood methods (Fan and Li, 2001) for function estimation and variable selection. In this section, we assume there are totally $p+q$ variables $\mathbf{X} = (x_1, \ldots, x_p, u_1, \ldots, u_q)$. The first p variables $x_1, \ldots, x_p$ are continuous, and the remaining q variables $u_1, \ldots, u_q$ are categorical (factors). Each u_j is coded to be dummy vector $\mathbf{z}_j$, $j = 1, \ldots, q$, and each u_j has d_j categories. Assume the response variable Y given $\mathbf{X} = \mathbf{x}$ follows a distribution from the exponential family as in (2.1).

2.3.1 Formulation

Given a random sample $\{y_i, x_{1i}, \ldots, x_{pi}, u_{1i}, \ldots, u_{qi}\}_{i=1}^n$ we consider the following model:

$$g(\mu_i) = \eta_i, \qquad \eta_i = \alpha + \sum_{j=1}^{p} f_j(x_{ji}) + \sum_{j=1}^{q} \mathbf{z}_{ji}^T \boldsymbol{\beta}_j, \tag{2.14}$$

where $\mu_i = E(Y_i|\mathbf{X}_i)$, $g(\cdot)$ is a known link function, α is the intercept, f_j, $j = 1, \ldots, p$, are unspecified functions, and $\boldsymbol{\beta}_j = (\beta_{j1}, \ldots, \beta_{jd_j})^T$. As in Section (2.1) , we assume f_j to be a univariate function. The density of $Y_i|\mathbf{X}_i$ is denoted by $p_i(y_i, \eta_i)$. Let $\ell_i = \log p_i(y_i, \eta_i)$ be the log-likelihood of y_i. Then a garrote-type estimator for GAMs is proposed by solving

$$\min_{\mathbf{c}} -\ell_n(\mathbf{c}) + \lambda \left(\sum_{j=1}^{p} c_j + \sum_{j=1}^{q} d_j c_{p+j} \right), \text{ subject to } c_j \geq 0, j = 1, \ldots, p+q, \tag{2.15}$$

where $\ell_n(\mathbf{c}) = \sum_{i=1}^{n} \ell_i(y_i, \tilde{\eta}_i)$, and

$$\tilde{\eta}_i = \hat{\alpha} + \sum_{j=1}^{p} c_j \hat{f}_j^{\text{init}}(x_{ji}) + \sum_{j=1}^{q} c_{j+p} \mathbf{z}_{ji}^T \hat{\boldsymbol{\beta}}_j^{\text{init}}, \tag{2.16}$$

and $\hat{f}_j^{\text{init}}(x_{ji})$, $j = 1, \ldots, p$ and $\hat{\boldsymbol{\beta}}_j^{\text{init}}$, $j = 1, \ldots, q$ are some initial estimates. The final garrote estimate of $f_j(x_{ji})$ is $\hat{c}_j \hat{f}_j^{\text{init}}(x_{ji}), j = 1, \ldots, p$ and the final garrote estimate of

$\hat{\boldsymbol{\beta}}_j$ is $\hat{c}_{j+p}\hat{\boldsymbol{\beta}}_j^{\text{init}}, j = 1, \ldots, q$. If some $\hat{c}_j = 0$, then the corresponding smoothing term or parametric term is dropped from the final model, which produces a sparse model. In theory, if both $\hat{f}_j^{\text{init}}, j = 1, \ldots, p$ and $\hat{\boldsymbol{\beta}}_j^{\text{init}}, j = 1, \ldots, q$ are consistent estimators, the garrote solution of our method may have oracle properties.

2.3.2 Computational Algorithms

In the context of generalized additive models, the garrote becomes a penalized likelihood problem instead of the penalized least squares. To solve the garrote in GAMs, we will use a Newton-Raphson iteration algorithm in the same spirit of the iteratively reweighted least squares (IRLS) (Tibshirani, 1997). At each iteration, we replace the weighted least squares by solving a weighted least squares garrote. In particular, define $\mathbf{c} = (c_1, \ldots, c_{p+q})^T$, the gradient $\mathbf{g} = -\partial \ell_n / \partial \mathbf{c}$, the Hessian matrix $\mathbf{H} = -\partial^2 \ell_n / \partial \mathbf{c}\mathbf{c}^T$. Let $\mathbf{W}$ be the Cholesky decomposition of $\mathbf{H}$ such that $\mathbf{H} = \mathbf{W}^T\mathbf{W}$. Define the working variable $\mathbf{v} = (\mathbf{W}^T)^{-1}(\mathbf{H}\mathbf{c} - \mathbf{g})$. Then a second-order Taylor expansion of $-\ell_n$ can be approximated by $\|\mathbf{v} - \mathbf{W}\mathbf{c}\|^2$ plus some constant not containing $\mathbf{c}$.

To get the garrote solution for GAMs in (2.15) for a fixed λ, we propose the following procedure:

1. Get the initial fits $\hat{f}_j^{\text{init}},\ j = 1, \ldots, p$ and $\hat{\boldsymbol{\beta}}_j^{\text{init}},\ j = 1, \ldots, q$.
2. Initialize $\hat{\mathbf{c}} = \mathbf{0}$.
3. Compute $\mathbf{g}$, $\mathbf{H}$, $\mathbf{W}$, and $\mathbf{v}$ based on the current value of $\hat{\mathbf{c}}$.
4. Minimize $\|\mathbf{v} - \mathbf{W}\mathbf{c}\|^2 + \lambda \left(\sum_{j=1}^{p} c_j + \sum_{j=1}^{q} d_j c_{p+j} \right)$, subject to $c_j \geq 0, j = 1, \ldots, p+q$ for given λ.
5. Repeat Steps 3 and 4 until the convergence criterion meets.

Breiman (1995) used the quadratic programming techniques to solve the original garrote problem, and developed a Fortran routine to implement this algorithm. We will adapt this garrote routine in Step 4 of our algorithm. Next, we will discuss two

key issues in the implementation of the proposed computational algorithms. One is the computation of initial estimates, and the other is the selection of the tuning parameter.

(1) Initial Estimation for GAMs

1. **Fitting GAMs with *mgcv* in R**
   There are several packages in R to fit the GAM. In particular, the function *gam* in the *gam* library implements the backfitting algorithm. Gu (2002) developed the *gss* package to fit GAM in the framework of smoothing splines. Both of these procedures perform well in practice, but they suffer from expensive computation due to choosing multiple smoothing parameters. If there are p smoothers, one has to search the optimal smoothing parameters in a p-dimensional space. Wood (2000,2004) suggested an efficient algorithm to estimate all components and automatically choose the smoothing parameters at the same time. He developed a package called *mgcv* library in R. Cantoni et al. (2006) tried several fitting methods and showed that Wood's package performs much better than other methods. In our approach , we employ the method of Wood (2000, 2004) and its package in R.

2. **When p is Large (relatively to n)** The package above works well in general except for data of small sample sizes. The *mgcv* package seems to be unstable and tends to experience convergence problems in practice when n is relatively small compared to p, say, $p = 10$, $n = 50$. In the following, we propose a boosting algorithm to fit the GAM by componentwise regression and smoothing based on the forward stagewise regression (Friedman et al., 2000), L_2 Boosting (Buhlmann and Yu, 2005) and likelihood-based boosting (Gerhard and Binder, 2006).

   We use the penalized thin-plate regression splines as weak learners. The loss function is the negative likelihood (deviance). Define $\mathbf{y} = (y_1, \ldots, y_n)^T$, $\mathbf{f}_j = (f_j(x_{j1}), \ldots, f_j(x_{jn}))^T$ and $\mathbf{1} = (1, \ldots, 1)^T$. The boosting procedure is as following:

(a) Fit the models with intercept only:

$$\begin{aligned}
\hat{\boldsymbol{\alpha}}^{(0)} &= \bar{y}\mathbf{1}, \\
\hat{\boldsymbol{\beta}}^{(0)} &= \mathbf{0}, \\
\hat{\mathbf{f}}_j^{(0)} &= \mathbf{0},\ j = 1, \ldots, p, \\
\hat{\boldsymbol{\eta}}^{(0)} &= (\bar{y}, \ldots, \bar{y})^T, \\
\hat{\boldsymbol{\mu}}^{(0)} &= g^{-1}(\hat{\boldsymbol{\eta}}^{(0)}).
\end{aligned}$$

(b) For $m = 0, 1, \ldots,$ do;

- **Fit parametric (linear) terms**

  Fit generalized linear models (GLM) with $(\mathbf{y} - \hat{\boldsymbol{\mu}}^{(m)})$ as response and $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_q)$ as covariates, and obtain $\hat{\alpha}$ and $\hat{\boldsymbol{\beta}}$. Update:

$$\begin{aligned}
\hat{\boldsymbol{\alpha}}^{(m+1)} &= \hat{\boldsymbol{\alpha}}^{(m)} + \hat{\alpha}\mathbf{1}, \\
\hat{\boldsymbol{\beta}}^{(m+1)} &= \hat{\boldsymbol{\beta}}^{(m)} + \hat{\boldsymbol{\beta}}, \\
\hat{\boldsymbol{\eta}}_{\text{temp}} &= \hat{\boldsymbol{\alpha}}^{(m+1)} + \mathbf{Z}\hat{\boldsymbol{\beta}}^{(m+1)} + \sum_{j=1}^{p} \mathbf{f}_j^{(m)}.
\end{aligned}$$

- **Fit smoothing terms**

  For $j = 1, \ldots, p,$ do;

  – Estimate $\hat{\mathbf{f}}_{j,\text{temp}}$ with $(\mathbf{y} - g^{-1}(\hat{\boldsymbol{\eta}}_{\text{temp}}))$ as response, using $gam(mgcv)$ in R, and set all the smoothing parameters to be large.

  – Let $\hat{\boldsymbol{\eta}}_{j,\text{temp}} = \hat{\boldsymbol{\eta}}_{\text{temp}} + \hat{\mathbf{f}}_{j,\text{temp}}$.

  end;

  Choose the κth term,

  where $\kappa = \max_j \big(\text{Deviance}(\hat{\boldsymbol{\eta}}_{\text{temp}}) - \text{Deviance}(\hat{\boldsymbol{\eta}}_{j,\text{temp}})\big)$.

  Update:

$$\begin{aligned}
\hat{\mathbf{f}}^{(m+1)} &= \hat{\mathbf{f}}^{(m)} + \hat{\mathbf{f}}_{\kappa,\text{temp}}, \\
\hat{\boldsymbol{\eta}}^{(m+1)} &= \hat{\boldsymbol{\eta}}_{\text{temp}} + \hat{\mathbf{f}}^{(m+1)}, \\
\hat{\boldsymbol{\mu}}^{(m+1)} &= g^{-1}(\hat{\boldsymbol{\eta}}^{(m+1)}).
\end{aligned}$$

  Update m to $m + 1$.

until the total deviance is small, end.

In practice, we set the smoothing parameters for component thin-plate splines to be 100, which performs well in our simulations. Based on our numerical experience, our boosting algorithm is able to provide a good initial estimate when the sample size is relatively small.

(2) Tuning Parameter Selection

To estimate the tuning parameter λ in (2.15), we use the BIC criterion:

$$\text{BIC}_\lambda = -2\ell_n(\hat{\mathbf{c}}) + \text{df}_\lambda \times \log n, \tag{2.17}$$

where $\hat{\mathbf{c}}$ is the garrote solution for given λ, and df_λ is the degree of freedom. We use the definition of Yuan and Lin (2006). For categorical terms, we have,

$$\text{df}_\lambda = 2\sum_{j=p+1}^{p+q} I(\hat{c}_j > 0) + \sum_{j=p+1}^{p+q} \hat{c}_j(d_{j-p} - 2). \tag{2.18}$$

For smoothing terms, the degree of freedom of the garrote is:

$$\text{df}_\lambda = 2\sum_{j=1}^{p} I(\hat{c}_j > 0) + \sum_{j=1}^{p} \hat{c}_j(\text{edf}_j - 2), \tag{2.19}$$

where edf_j is the effective degree of freedom defined as the trace of influence matrix. In the GAM fitting procedures of Section (2.1.2), the total influence matrix is defined as $\mathbf{H} = (\mathbf{B}^T\mathbf{B} + \sum_{j=1}^{p} \mathbf{\Omega}_j)^{-1}\mathbf{B}^T$.

2.4 Simulation Studies

We investigate the performance of our proposed methods in three examples for semi-parametric additive models We compare our procedures with other commonly used model selection methods: MARS and COSSO. For each example, we start with the description of the simulation design, which is followed by the summary of results.

2.4.1 Example 1

Consider a Gaussian additive model with $p = 20$ covariates. The first ten covariates are categorical, each with three levels. The second ten covariates are continuous with certain degree of correlation.

First, we generate $Q_1, \ldots, Q_{20}$ and U independently from $\mathrm{U}(0,1)$. Define $X_j = Q_j$, $j = 1, \ldots, 10$. Then these 10 predictors are trichotomized as 0, 1 or 2 depending on whether they are smaller than $\frac{1}{3}$, larger than $\frac{2}{3}$ or in between. $X_j, j = 11, \ldots, 20$ are generated as $X_j = (Q_j + tU)/(1+t)$. This construction makes ten continuous covariates be correlated with a compound symmetry (CS) covariance structure: $\mathrm{corr}(X_j, X_k) = t^2/(1+t^2)$ for any pair $j \neq k$ and $j, k = 11, \ldots, 20$. The parameter t controls the degree of correlation: no correlation, moderate correlation and strong correlation. We use $t = 0, 1, 3$, which result in the $\mathrm{corr}(X_j, X_k) = 0, 0.5, 0.9, j \neq k$ respectively.

The true model is:

$$\begin{aligned} Y &= 2.5I(X_1 = 1) + I(X_1 = 2) + 3I(X_2 = 1) + 1.5I(X_2 = 2) + \\ &\quad 5f_1(X_{11}) + 3f_2(X_{12}) + 4f_3(X_{13}) + 6f_4(X_{14}) + \epsilon, \end{aligned} \tag{2.20}$$

where

$$f_1(s) = s;\ f_2(s) = (2s-1)^2;\ f_3(s) = \frac{\sin(2\pi s)}{2 - \sin(2\pi s)}; \tag{2.21}$$

$$f_4(s) = 0.1\sin(2\pi s) + 0.2\cos(2\pi s) + 0.3\sin^2(2\pi s) + 0.4\cos^3(2\pi s) + 0.5\sin^3(2\pi s). \tag{2.22}$$

The error ϵ is iid from a centered normal distribution with $\sigma^2 = 3.5$, which yields a signal to noise ratio of 3. For this model, there are totally 14 uninformative variables. We choose the sample size to be 250. For each model setting, 100 data sets were generated. To measure the prediction accuracy, we use the integrated squared error (ISE), which is defined as $E_{\mathbf{X}}((\hat{\eta} - \eta)^2)$. The average ISE is estimated by Monte Carlo using 2000 test points.

We compare our garrote method with MARS. Table 2.1 displays the selection frequency of two dummies (1 and 2, out of 100 simulations) for each categorical variable. Table 2.2 shows the selection frequency of each continuous variable in the

Table 2.1: Frequency of selected dummies for categorical variables in Example 1.

t	Methods	X_1	X_2	X_3	X_4	X_5
0	garrote	100\100	100\100	3\3	3\3	3\3
	MARS	100\90	100\100	4\3	3\5	5\5
1	garrote	100\100	100\100	3\3	2\2	3\3
	MARS	100\58	100\90	3\5	7\6	5\12
3	garrote	100\100	100\100	7\7	9\9	5\5
	MARS	100\37	100\57	9\10	7\12	10\7
t	Methods	X_6	X_7	X_8	X_9	X_{10}
0	garrote	5\5	5\5	5\5	4\4	2\2
	MARS	6\4	6\3	6\6	3\5	3\3
1	garrote	7\7	5\5	4\4	7\7	2\2
	MARS	9\9	5\12	5\12	8\10	7\7
3	garrote	8\8	9\9	7\7	9\9	3\3
	MARS	11\9	10\13	4\5	11\14	9\6

The number before backslash is the selection frequency of level "1", and the number after backslash is for level "2".

final models. Generally, MARS overall chooses uninformative variables more often. We can see our method tends to choose a sparse model and removes noisy variables more effectively than MARS. In high correlation settings ($t = 1, 3$), we notice that the garrote drops slightly more important variables than MARS. The garrote method has overall smaller average ISEs than MARS.

2.4.2 Example 2

In this example, we compare our garrote method with *gam*(*mgcv*) (ggam), GAM with boosting (gBoostgam) and COSSO in logistic additive models. All the simulation settings are similar to Zhang and Lin (2006). Since the garrote with *gam*(*mgcv*) is computationally unstable when the sample size is small, we do not include this method in some of the simulation settings.

To be consistent with Zhang and Lin (2006) we measure the estimation accuracy via two criteria: the comparative Kullback-Leibler distance (CKL) between $\boldsymbol{\eta}$ and $\hat{\boldsymbol{\eta}}$ and the expected misclassification rate (EMR). For logistic additive models, CKL is

Table 2.2: Frequency of selected continuous variables in Example 1.

t	Methods	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}	X_{17}	X_{18}	X_{19}	X_{20}
0	garrote	100	100	100	100	10	10	11	10	12	10
	MARS	100	100	100	100	21	26	26	19	25	23
1	garrote	100	92	100	100	12	13	11	10	13	14
	MARS	100	99	100	100	25	18	30	24	22	25
3	garrote	87	83	100	100	11	14	9	10	11	12
	MARS	93	90	100	100	33	30	35	35	33	31

Table 2.3: The average ISE results in Example 1.

	t=0	t=1	t=3
garrote	0.37	0.41	0.46
MARS	0.76	0.59	0.67
oracle	0.27	0.26	0.25

The range of the standard errors for average ISE is 0.01 to 0.02 in all three settings.

defined as:

$$\mathrm{CKL}(\boldsymbol{\eta}, \hat{\boldsymbol{\eta}}) = \frac{1}{n}\sum_{i=1}^{n}\left(\log(1+\exp(\hat{\eta}_i)) - \mu_i\hat{\eta}_i\right).$$

We use $10,000$ testing points to estimate μ. The same testing data are also used to calculate EMR. The Bayes error for each simulation scenario is reported. We simulate 100 data sets and report the means of CKL and EMR. We also report the selection frequency of each factor and/or variable in the final model.

Consider a logistic additive model in which the covariates are correlated. The data and the true logit function are the same as in Example 1, except we only consider continuous variables here. First, we generate $Q_1, \ldots, Q_{10}$ and U independently from $U(0,1)$. $X_j, j = 1, \ldots, 10$ are built according to "compound symmetry " design: $X_j = (Q_j + tU)/(1+t)$ with $\mathrm{corr}(X_j, X_k) = t^2/(1+t^2)$ for any pair $j \neq k$. We use $t = 0, 1$, which result in the $\mathrm{corr}(X_j, X_k) = 0, 0.5, j \neq k$ respectively.

The true logit function is:

$$\eta(\mathbf{X}) = 5f_1(X_1) + 3f_2(X_2) + 4f_3(X_3) + 6f_4(X_4), \tag{2.23}$$

Table 2.4: Frequency of the selected variables in Example 2 ($t = 0$).

n	Methods	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
100	ggam	–	–	–	–	–	–	–	–	–	–
	gBoostgam	81	38	91	100	14	16	23	14	17	11
	COSSO	86	32	99	95	32	30	29	30	31	29
200	ggam	–	–	–	–	–	–	–	–	–	–
	gBoostgam	99	59	100	100	6	10	10	7	10	7
	COSSO	95	40	100	100	10	14	13	7	12	12
500	ggam	100	98	100	100	2	2	1	0	1	0
	gBoostgam	100	78	100	100	1	0	0	2	1	1
	COSSO	100	85	100	100	6	8	6	8	7	3

where

$$f_1(s) = s;\ f_2(s) = (2s-1)^2;\ f_3(s) = \frac{\sin(2\pi s)}{2-\sin(2\pi s)}; \tag{2.24}$$

$$f_4(s) = 0.1\sin(2\pi s) + 0.2\cos(2\pi s) + 0.3\sin^2(2\pi s) + 0.4\cos^3(2\pi s) + 0.5\sin^3(2\pi s). \tag{2.25}$$

y_i is generated independently from $\text{Bin}(1, \exp(\eta_i)/(1+\exp(\eta_i)))$. We try three sample sizes: $100, 200, 500$. The Bayes error is 0.134 for $t = 0$ and 0.142 for $t = 1$.

In Figure 2.1, we plot the total deviance against the number of boosting steps when $n = 100, t = 0$. Since the deviance is relative small when the boosting iteration M is greater than 200, we try several iterations: $M = 300, 400, 500$. Our results show that boosting with 400 iterations gives the best performance in this example. The typical initial estimates for the component functions in one data set are plotted in Figure 2.2. We can see the fitted curves overall capture the true function forms well.

In Figure 2.3, we plot the estimated individual functions given by our method and the three broken lines respectively correspond the 10th, 50th and 90th percentiles of the estimated EMR among the 100 simulations. To find the sampling variability of the garrote estimates at each point, we plot the 2.5th, 50th and 97.5th percentiles of the estimated functions at 100 random data points among 100 simulations in Figure 2.4. This forms a 95% pointwise empirical confidence envelope for the garrote estimates. These results show that our method can capture the true function except at boundaries, where the data points are scarce.

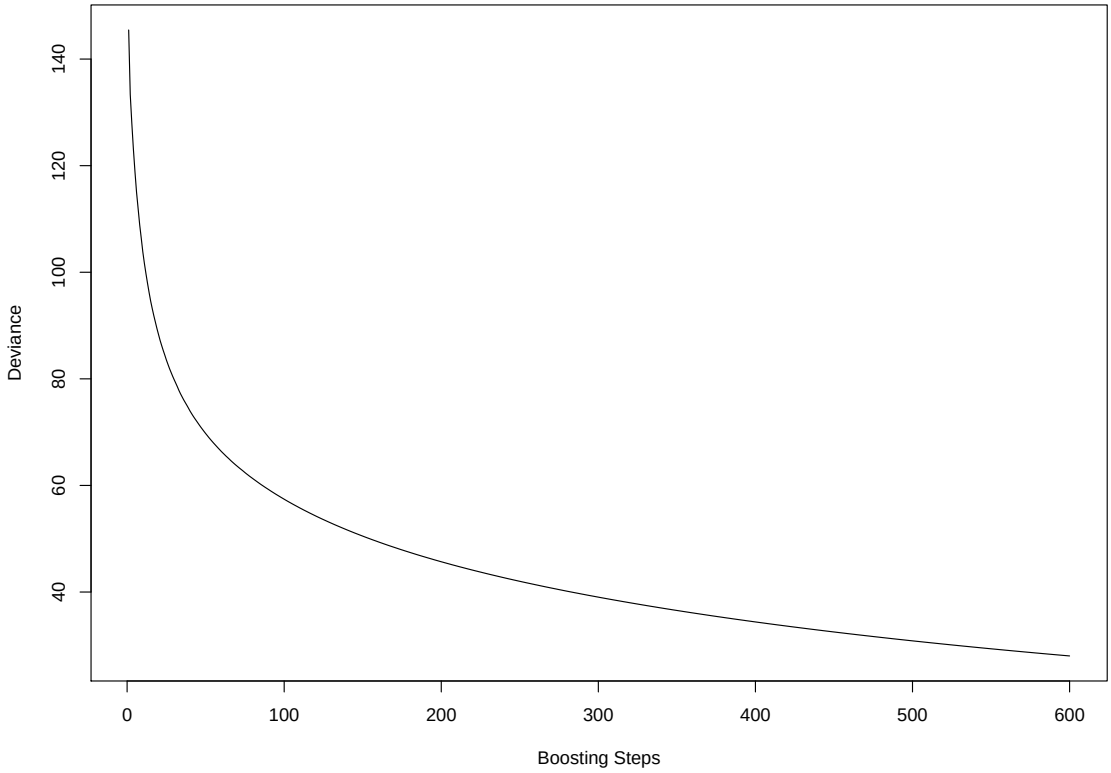

Figure 2.1: Total deviance versus the number of boosting steps in Example 2 ($n = 100, t = 0$).

Table 2.5: Average CKL and EMR in Example 2 ($t = 0$).

n	Methods	CKL	EMR
100	ggam	–	–
	gBoostgam	0.27	0.18
	COSSO	0.46	0.22
200	ggam	–	–
	gBoostgam	0.23	0.16
	COSSO	0.37	0.17
500	ggam	0.20	0.14
	gBoostgam	0.21	0.14
	COSSO	0.33	0.15

The range of the standard errors for average CKL is 0.01 to 0.05, for average EMR is 0.01 to 0.03.

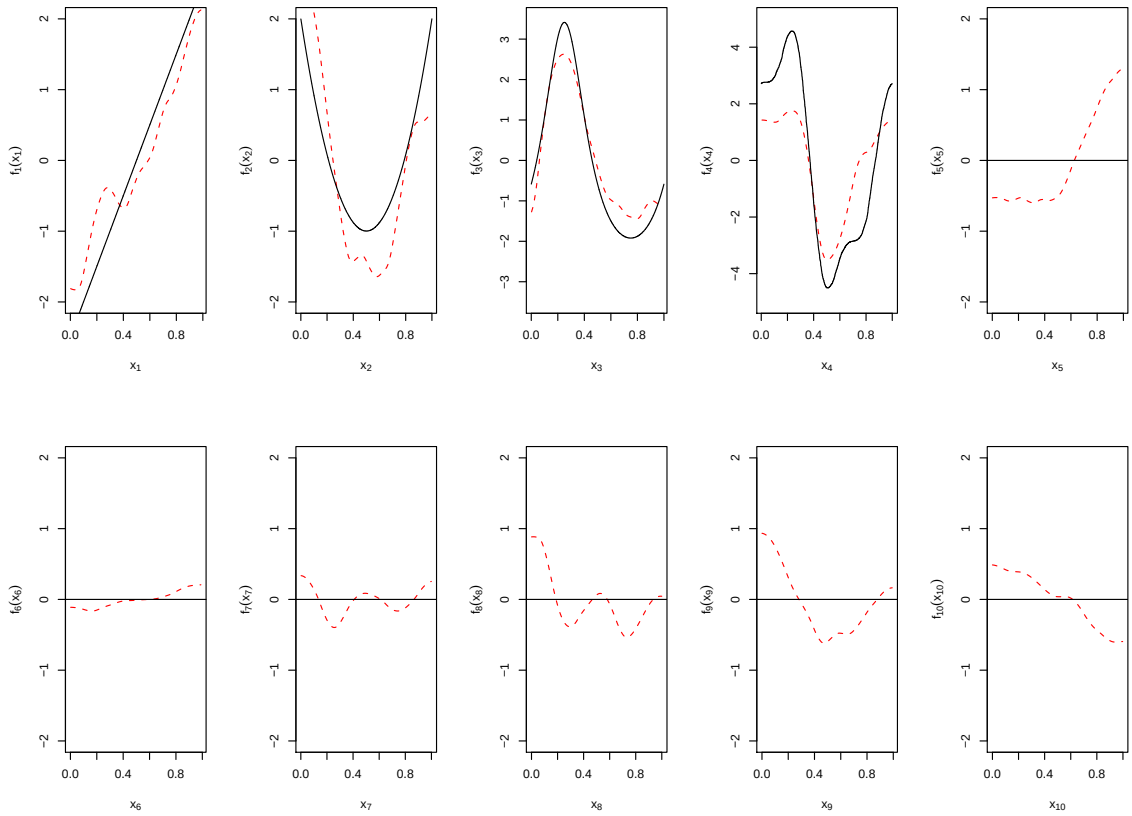

Figure 2.2: Initial estimates for component functions with the boosting algorithm in Example 2 ($n = 100, t = 0$). Black solid lines indicate the true functions, red dashed lines are estimated functions with boosting. The boosting iteration is $M = 400$.

Table 2.6: Frequency of the selected variables in Example 2 ($t = 1$).

n	Methods	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
100	ggam	–	–	–	–	–	–	–	–	–	–
	gBoostgam	77	32	100	100	21	16	21	15	13	18
	COSSO	74	38	100	81	38	33	30	33	34	35
200	ggam	–	–	–	–	–	–	–	–	–	–
	gBoostgam	91	40	100	100	7	5	15	7	9	10
	COSSO	84	35	100	100	8	17	12	13	15	13
500	ggam	99	68	100	100	3	1	2	1	3	4
	gBoostgam	100	76	100	100	5	4	3	3	3	7
	COSSO	100	77	100	100	14	9	12	7	11	14

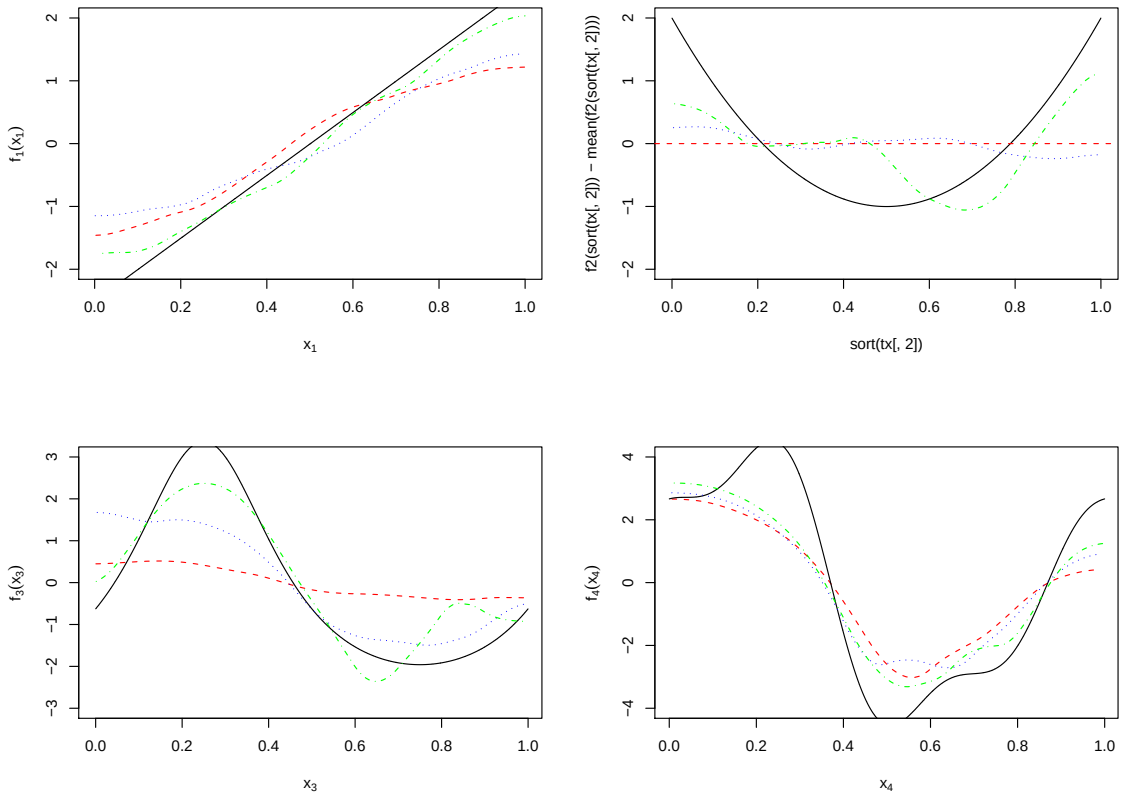

Figure 2.3: Estimated functions with garrote in Example 2 ($n = 100, t = 0$). Red dashed lines indicate the 10th best (in terms of misclassification rate), blue dotted lines indicate the 50th best, and green dot-dashed lines are the 90th best. The black solid lines are true function curves.

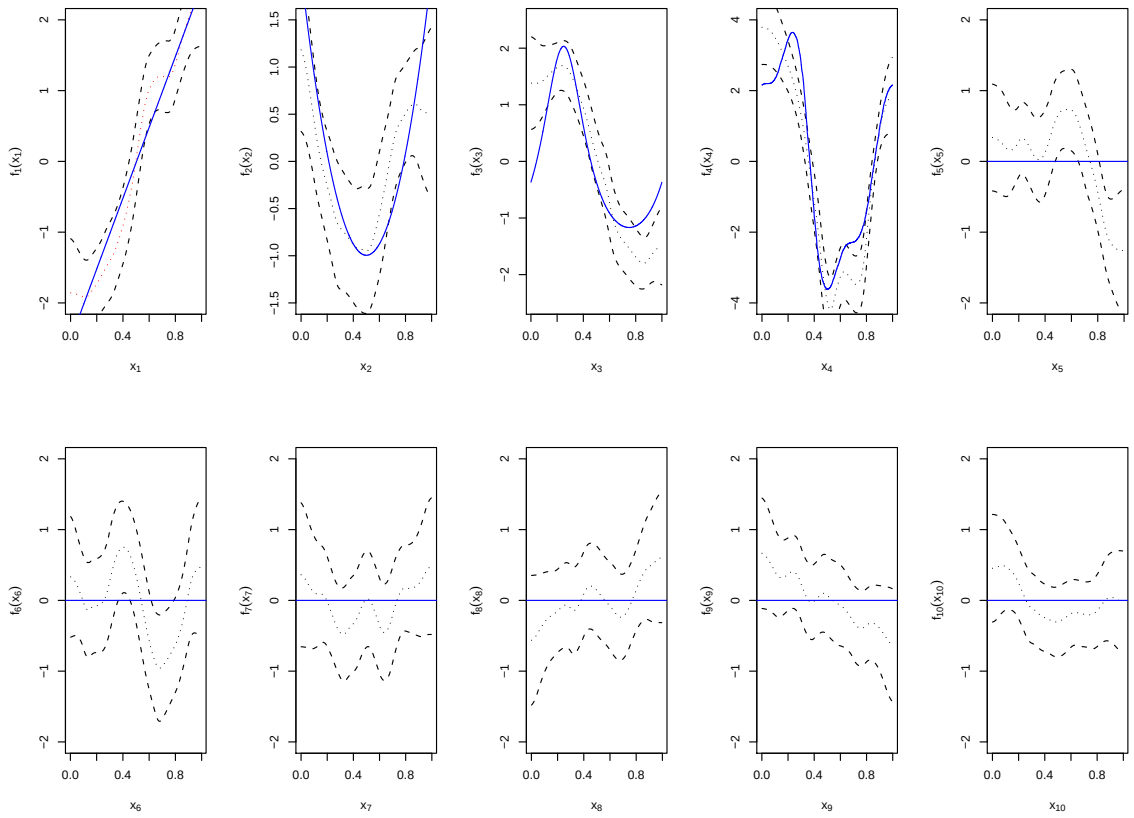

Figure 2.4: Sampling variability of the estimated functions with garrote at each point based on 100 simulations in Example 2 ($n = 100, t = 0$). Dashed lines are the 2.5th and 97.5th percentiles of 100 simulations, and they form the 95% envelope. Dotted lines are the 50th percentiles. Blue solid lines are true function curves.

Table 2.7: Average CKL and EMR in Example 2 ($t = 1$).

n	Methods	CKL	EMR
100	ggam	– (–)	– (–)
	gBoostgam	0.39	0.19
	COSSO	0.47	0.22
200	ggam	–	–
	gBoostgam	0.35	0.17
	COSSO	0.38	0.18
500	ggam	0.33	0.15
	gBoostgam	0.34	0.15
	COSSO	0.34	0.16

The range of the standard errors for average CKL is 0.01 to 0.05, for average EMR is 0.01 to 0.02.

Tables 2.4, 2.5, 2.6, 2.7 summarize the results for Example 2 with $t = 0$ and $t = 1$ respectively. The missing values indicate that the algorithm for $gam(mgcv)$ does not converge. For COSSO, we use the algorithm with basis= 50. We can see the garrote with boosting gives satisfactory performance even when the sample size is relatively small compared to the data dimension. In terms of variable selection, our method selects important variables more often than COSSO, and the final model is simpler than COSSO. The garrote with boosting also reduces CKL and EMR substantially. For large data ($n = 500$), the garrote with gam has competitive performance in both variable selection and estimation accuracy.

2.4.3 Example 3

In this example, we extend Example 2 and consider the semi-parametric GAM models where both continuous and categorical covariates are present. We generate $X_1, \ldots, X_{11}$ from $\mathrm{U}(0, 1)$, the first seven variables are generated in the same way as in Example 2. We then make the last four categorical variables by the following

Table 2.8: Variable frequency of group garrote with boosting in Example 3.

t	n	X_1	X_2	X_3	X_4	X_5	X_6	X_7	z_1	z_2	z_3	z_4
0	150	99	56	99	100	9	7	14	72	14	87	23
	200	100	66	99	100	3	2	5	73	10	97	5
	500	100	90	100	100	2	1	0	97	1	100	2
1	150	80	46	99	100	29	20	13	82	19	98	17
	200	85	50	100	100	16	10	6	91	15	99	8
	500	98	78	100	100	7	3	6	100	3	100	2

Table 2.9: Average CKL and EMR of the group garrote in Example 3.

t	n	CKL	EMR
0	150	0.35	0.16
	200	0.32	0.14
	500	0.28	0.12
1	150	0.40	0.19
	200	0.39	0.18
	500	0.34	0.16

The range of the standard errors for average CKL is 0.01 to 0.03, for average EMR is 0.01 to 0.02.

transformations:

$$
\begin{aligned}
z_1 &= I(X_8 \geq \frac{2}{3}) + 2I(X_8 \in [\frac{1}{3}, \frac{2}{3})) + I(X_8 < \frac{1}{3}), \\
z_2 &= I(X_9 \geq \frac{2}{3}) + I(X_9 \in [\frac{1}{3}, \frac{2}{3})) + I(X_9 < \frac{1}{3}), \\
z_3 &= 0.5I(X_{10} < \frac{1}{4}) + 1.5I(X_{10} \in [\frac{1}{4}, \frac{1}{2})) + 3I(X_{10} \in [\frac{1}{2}, \frac{3}{4})) + I(X_{10} \geq \frac{3}{4}), \\
z_4 &= I(X_{11} \geq \frac{2}{3}) + I(X_{11} \in [\frac{1}{3}, \frac{2}{3})) + I(X_{11} < \frac{1}{3}),
\end{aligned}
$$

where I is an indicator function. The true logit function is ,

$$\eta = -4.5 + 5f_1(X_1) + 3f_2(X_2) + 4f_3(X_3) + 6f_4(X_4) + z_1 + z_3.$$

We consider $t = 0, 1$ and sample size $n = 150, 200, 500$. The Bayes error is 0.098 for $t = 0$, and 0.132 for $t = 1$. From Tables 2.8, 2.9, we can see our method gives smaller CKL and EMR as the sample size increases, and it produces sparse models with the error rate close to the Bayes rate when $n = 500$.

2.5 Real Examples

2.5.1 German Credit Data

The German credit data comes from a German bank and is provided by Professor Hans Hofmann in Strathclyde University (available at ftp.ics.uci.edu/pub/machine-learning-databases/statlog/). The data has 20 covariates and 1000 observations which represent 1000 past credit applicants. Each applicant was rated as "good credit" (700 cases, coded as "1") or "bad credit" (300 cases, coded as "0"). We want to obtain a model for the credit scoring rule that can be used to determine if a new applicant presents a good or bad credit risk, based on values for one or more of the predictor variables.

Various statistical models and data mining tools, such as linear discriminant analysis, logistic regression, classification tree, neural network, boosting and support vector machines (SVM) have been used to evaluate the credit worthiness of potential borrowers in order to reduce the default risk (Franke, Hardle, and Stahl, 2000; Shao 2004; Rätsch, *et al.*, 2000). In many applications of credit scoring, important variables are selected to get good prediction and interpretation of the final model. In this data,we have 7 numerical predictors and 13 categorical predictors (variable annotations are given in Appendix). The number of levels of these 13 categorical variables ranges from 2 to 10. Most of these categorical variables are derived from numerical variables by score developers, for purpose of easy interpretation and implementation. For example, the variable "present employment since" represents the number of years the applicant was at the present job. This original numerical variable was divided into five bins: unemployed, < 1 year, $[1, 4)$ years, $[4, 7)$ years and ≥ 7 years. In practice, score developers would like to keep or drop all levels simultaneously in the process of variable selection. However, most of current variable selection and statistical models can not achieve this. Our garrote method for generalized additive models can select grouped categorical variables and continuous variable at the same time. We apply our method to the data and compare our results with several other benchmark methods in machine learning.

In Table 2.10, the average misclassification error (with standard error) over 100 partitions of the data sets is given. In each partition, we divide the 1000 observations into two sets: training set (size of 700) and test set (size of 300). We apply our garrote method and other methods to the training set, and then use the test set to calculate the misclassification error. We observe that our method has the smallest misclassification error among all methods.

Table 2.10: The average error in German credit data.

	Misclassification error
SVM with RBF-Kernel	23.61 (0.21)
Kernel Fisher Discriminant	23.71 (0.22)
AdaBoost with RBF-Network	27.45 (0.25)
garrote	22.54 (0.22)

All other methods use all of 20 variables to fit the final model, while our method results in a much smaller and simpler model. The average model size of our garrote models is 10.52. When we apply our method to the complete data set (1000 observations), it selects 9 variables : status of existing checking account (x_1), duration in month (x_2), credit history (x_3), loan purpose (x_4), credit amount (x_5), savings account/bonds (x_6), present employment since (x_7), other debtors / guarantors (x_{10}). Among these 9 variables in the final model, x_2 and x_5 are continuous variables. We plot the garrote estimates for these two variables in Figure 2.5. It shows the variable "duration in month" has a linear trend, while a nonlinear trend exists in the other variable "credit amount" .

2.5.2 Wisconsin Prognostic Breast Cancer (wpbc) Data

The wpbc data is from the UCI repository (Blake and Merx, 1998). The data is collected by studying recurrent events in breast cancer patients based on their digital image of breast tissues. In this study, there are 194 patients which have complete records. For each patient, 32 continuous covariates are derived by computing the characteristics of breast cell nuclei presenting in the image. The annotations of these 32 variables are in Appendix.

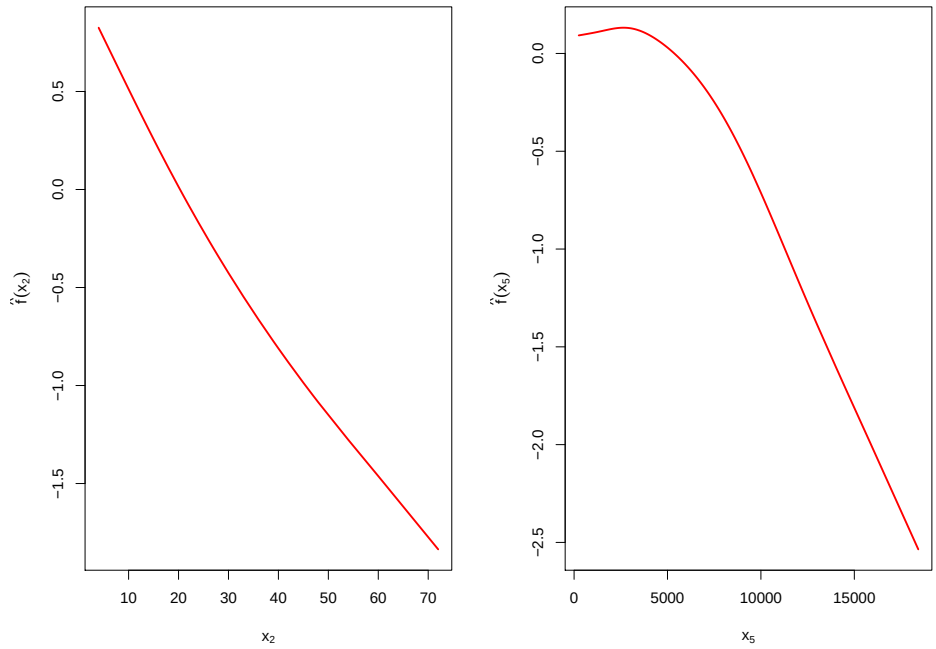

Figure 2.5: Garrote estimates of component functions for German credit data.

Instead of investigating survival time of recurrent events, we focus on a binary classification problem: recurrence vs. non-recurrence. Since we are primarily interested in the relationship between the log odds ratio of recurrence to non-recurrence and prediction covariates, the generalized additive model is a good choice. Since the number of observations ($n = 194$) is relatively small to the dimension of covariates ($p = 32$), variable selection will be an issue for most nonparametric regressions. Our garrote method can deal with this problem efficiently. We apply our boosting algorithm to get initial estimates with 32 variables first, and then use the garrote to select important covariates.

In Table 2.11, the 5-fold cross validation misclassification errors (with standard error) for three methods are given. We compare our method with the stepwise GLM and MARS. Final results suggest that our new method can outperform MARS by a good margin.

Table 2.11: Five-fold cross validation error in wpbc data.

	5-fold cross validation error ($\times 100$)
Stepwise GLM	30.21 (2.52)
MARS	27.18 (2.71)
two-stage garrote	24.18 (2.24)

We also fit the complete data set ($n = 194$) with MARS and the garrote. As a reference, we fit a logistic regression with stepwise selection for complete data. The logistic regression with stepwise selects 19 variables: mean radius, mean texture, mean smoothness, mean symmetry, mean fractaldim, SE texture, SE perimeter, SE compactness, SE concavity, SE concavepoints, SE symmetry, SE fractaldim, worst radius, worst perimeter, worst area, worst smoothness, worst compactness, tsize, pnodes. MARS selects 6 variables: mean fractaldim, SE radius, SE perimeter, worst perimeter worst smoothness and pnodes. Our method selects 8 variables in the final model: tsize, worst radius, pnodes, worst smoothness, SE texture, SE perimeter, mean symmetry and SE concavity. The variables selected by our methods is a subset of the stepwise regression, but quite different from the variables selected by MARS except for only one variable: worst smoothness. In Figures 2.6 and 2.7, the estimated component functions selected by MARS and the garrote are plotted. For all the selected variables in both methods, the model fit suggests they have very strong nonlinear trends. We can see the MARS fit is not as smooth as the fitted components given by our garrote method.

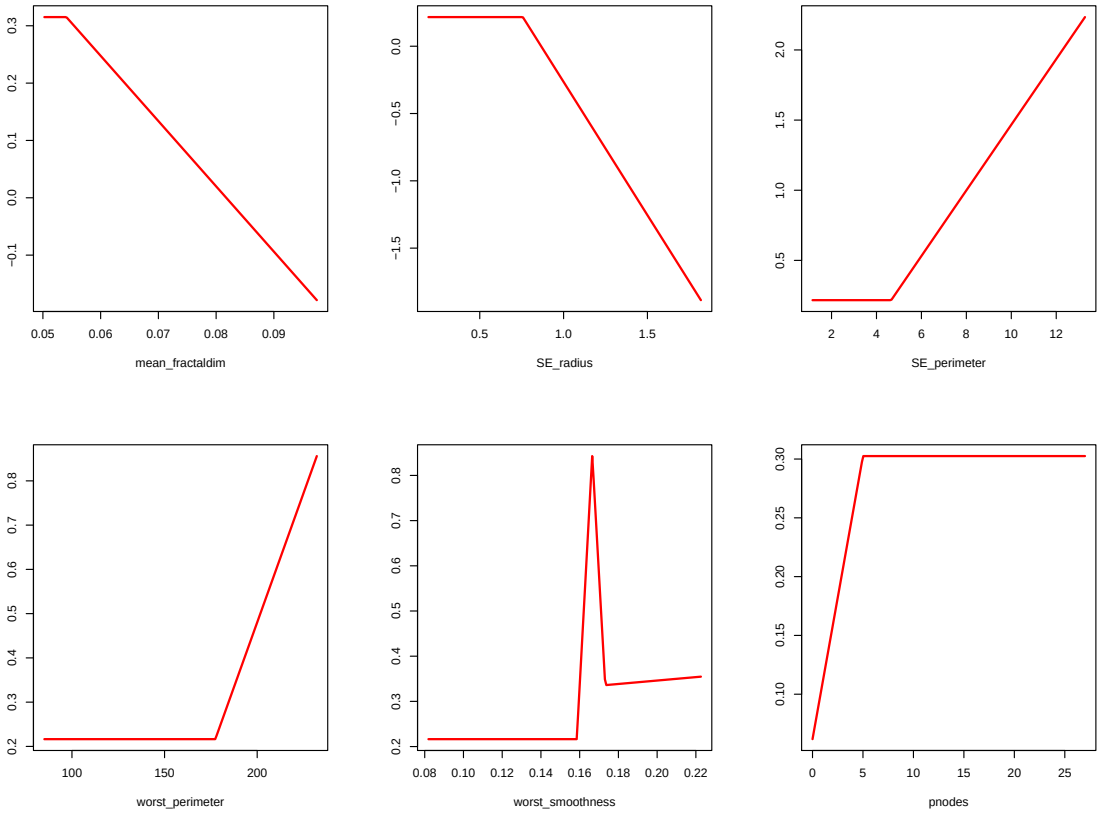

Figure 2.6: MARS fitting for selected component functions in wpbc data.

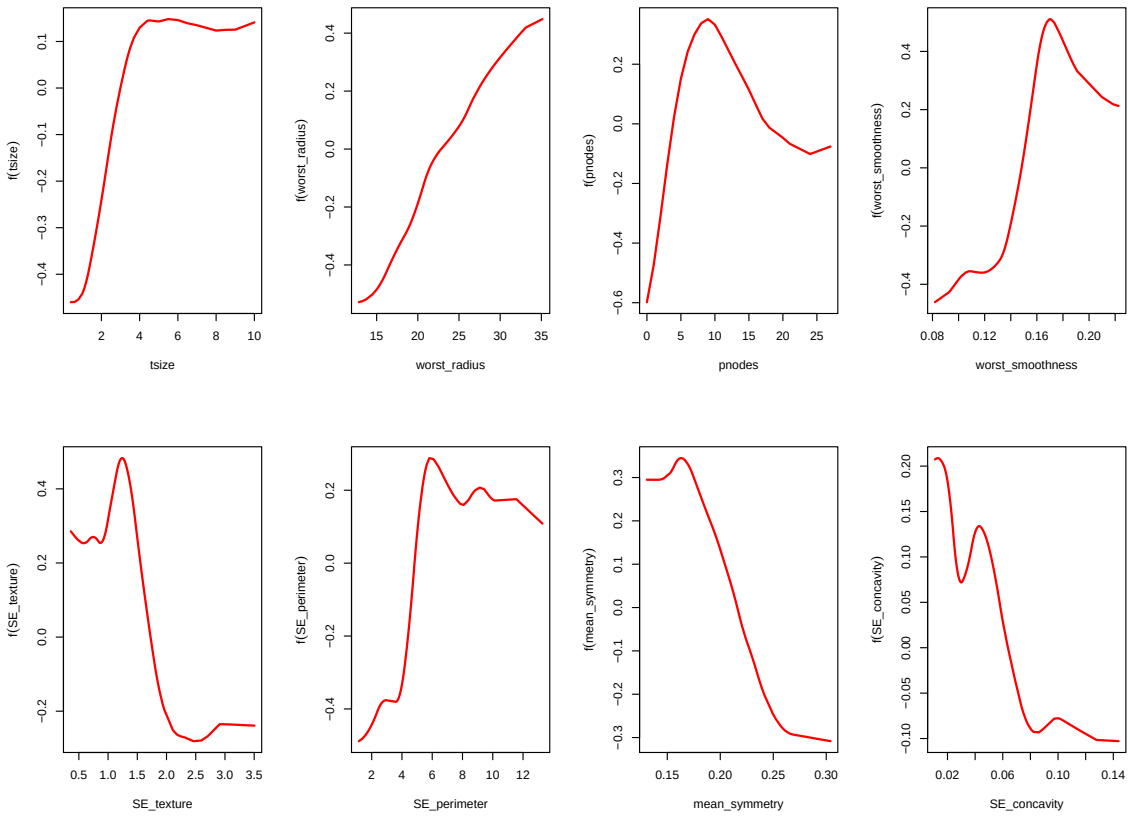

Figure 2.7: Garrote estimates for selected component functions in wpbc data.

Chapter 3

New Variable Selection for Additive Cox Models

In this chapter, we first review the additive Cox model and existing methods of variable selection in this model. Then we propose a garrote method for variable selection in the penalized partial likelihood of additive Cox models. A new algorithm is suggested to get reliable initial estimates for the additive Cox models. The computation and tuning parameter issues are addressed as well. The simulations and real examples are then presented.

3.1 Background Review

3.1.1 Additive Cox Models

The purpose of survival data analysis is to study the dependence of survival time T on the p-dimensional predictors $\mathbf{X} = (X_1, \ldots, X_p)$. The classical proportional hazards model, also known as the Cox model, assumes that,

$$h(T|\mathbf{X}) = h_0(T) \exp(\sum_{j=1}^{p} X_j \beta_j), \tag{3.1}$$

where $h(T|\mathbf{X})$ is the hazard function at time T given covariates $\mathbf{X} = (X_1, \ldots, X_p)$, $h_0(T)$ is an unspecified baseline hazard function, and β_j, $j = 1, \ldots, p$ are regression

coefficients.

Suppose a typical survival data of n samples is given. Let T be the failure time, C be the censoring time, and $\tilde{T} = \min(T, C)$, $\delta = I(T \leq C)$. Assume that T and C are conditionally independent given $\mathbf{X}$, and the censoring is noninformative. For simplicity, we assume there are no ties in the observed failure times. Then the log partial likelihood of the Cox model is:

$$\ell_i(\eta_i) \equiv \delta_i \left(\eta_i - \log \left[\sum_{j=1}^{n} I(\tilde{T}_j \geq \tilde{T}_i) \exp(\eta_i) \right] \right), i = 1, \ldots, n. \tag{3.2}$$

In the parametric Cox model, we have

$$\eta_i = \sum_{j=1}^{p} X_{ji}\beta_j, i = 1, \ldots, n. \tag{3.3}$$

To increase the model flexibility, it is natural to extend the model (3.1) to a nonparametric additive Cox model as

$$h(T|\mathbf{X}) = h_0(T) \exp(\sum_{j=1}^{p} f_j(X_j)), \tag{3.4}$$

where f_j, $j = 1, \ldots, p$ are unspecified univariate smooth functions.

Hastie and Tibshirani (1990) proposed a computation algorithm based on the backfitting to estimate the components functions in (3.4). They use the Newton-Raphson algorithm and the additive Gauss-Seidel procedure to update the nonlinear components iteratively. Since their method is based on the backfitting, the problem of choosing multiple smoothing parameters can be a challenging issue as in other backfitting additive models.

3.1.2 COSSO-type Variable Selection for Additive Cox Models

Due to the nature of censored data, it is very difficult to conduct nonparametric estimation and variable selection jointly in survival data. Very few methods are reported. Leng and Zhang (2007) extended the Component Selection and Smoothing

Operator (COSSO) to survival data. The COSSO penalty is a functional analogue of ℓ_1 penalty used in the lasso. With COSSO penalty, this method can result in a sparse estimated model.

In COSSO-type method, the nonparametric estimation of $\eta(\mathbf{X})$ is estimated via the minimization of a penalized partial likelihood (Leng and Zhang, 2007),

$$\min_{\eta\in\mathcal{H}} -\frac{1}{n}\sum_{i=1}^{n}\ell_i(\eta_i) + \tau J(\eta) + \lambda\sum_{\alpha=1}^{p}\theta_\alpha,\ s.t.\ \theta_\alpha \geq 0, \tag{3.5}$$

where $J(\eta) = \sum_{\alpha=1}^{p}\theta_\alpha^{-1}\left\|P^\alpha\eta\right\|^2$ is a roughness penalty and $P^\alpha\eta$ is the projection of η onto $\mathcal{H}_\alpha$, $\alpha = 1,\ldots,p$, which are orthogonal reproducing kernel Hilbert spaces. Here λ is smoothing parameter which controls the goodness of fit and sparsity in the solution.

Leng and Zhang (2007) conducted extensive experiments and showed that the COSSO-type method can do variable selection and component estimation simultaneously in nonparametric Cox models. Their results showed the COSSO-type method performs well even for highly censored survival data.

3.2 New Methodology for Additive Cox Models

3.2.1 New Methodology

Assume there are totally $p+q$ variables $\mathbf{X} = (x_1,\ldots,x_p,u_1,\ldots,u_q)$. The first p variables $x_1,\ldots,x_p$ are continuous, and the remaining q variables $u_1,\ldots,u_q$ are categorical (factors). Each u_j is coded to be a dummy vector $\mathbf{z}_j$, $j = 1,\ldots,q$, and each u_j has d_j categories.

We consider the log partial likelihood of the additive Cox model:

$$\ell_i(\eta_i) \equiv \delta_i\left(\eta_i - \log\left[\sum_{j=1}^{n} I(\tilde{T}_j \geq \tilde{T}_i)\exp(\eta_i)\right]\right), i = 1,\ldots,n, \tag{3.6}$$

where

$$\eta_i = \sum_{j=1}^{p} f_j(x_{ji}) + \sum_{j=1}^{q}\mathbf{z}_{ji}^T\boldsymbol{\beta}_j, i = 1,\ldots,n. \tag{3.7}$$

We now propose a garrote-type estimator for additive Cox model by solving

$$\min_{\mathbf{c}} -\ell_n(\mathbf{c}) + \lambda \left(\sum_{j=1}^{p} c_j + \sum_{j=1}^{q} d_j c_{p+j} \right), \text{ subject to } c_j \geq 0, j = 1, \ldots, p+q, \quad (3.8)$$

where $\ell_n(\mathbf{c}) = \sum_{i=1}^{n} \ell_i(y_i, \tilde{\eta}_i)$, and

$$\tilde{\eta}_i = \sum_{j=1}^{p} c_j \hat{f}_j^{\text{init}}(x_{ji}) + \sum_{j=1}^{q} c_{j+p} \mathbf{z}_{ji}^T \hat{\boldsymbol{\beta}}_j^{\text{init}}, \quad (3.9)$$

and $\hat{f}_j^{\text{init}}(x_{ji}),\ j = 1, \ldots, p$ and $\hat{\boldsymbol{\beta}}_j^{\text{init}},\ j = 1, \ldots, q$ are initial estimates. The final garrote estimate of $f_j(x_{ji})$ is $\hat{c}_j \hat{f}_j^{\text{init}}(x_{ji}), j = 1, \ldots, p$ and the final garrote estimate of $\hat{\boldsymbol{\beta}}_p$ is $\hat{c}_{j+p} \hat{\boldsymbol{\beta}}_j^{\text{init}}, j = 1, \ldots, q$. If some of $\hat{c}_j = 0$, $j = 1, \ldots, p+q$, then the corresponding smoothing term or parametric term is dropped from the final model, which produces a sparse model. In theory, if both $\hat{f}_j^{\text{init}}, j = 1, \ldots, p$ and $\hat{\boldsymbol{\beta}}_j^{\text{init}}, j = 1, \ldots, q$ are consistent estimators, the garrote solution of the additive Cox model may have oracle properties.

3.2.2 Computational Algorithms

In the additive Cox models, the garrote becomes a penalized partial likelihood problem. To solve the garrote, we propose a new algorithm by modifying the iteratively reweighted least squares. At each iteration, we replace the weighted least squares by solving a weighted least squares garrote. Define $\mathbf{c} = (c_1, \ldots, c_{p+q})^T$, the vector gradient $\mathbf{g} = -\partial \ell_n / \partial \mathbf{c}$, the Hessian matrix $\mathbf{H} = -\partial^2 \ell_n / \partial \mathbf{c} \mathbf{c}^T$, where ℓ_n is a partial likelihood function defined in (3.6). Let $\mathbf{W}$ be the Cholesky decomposition of $\mathbf{H}$ such that $\mathbf{H} = \mathbf{W}^T \mathbf{W}$. Define the working variable $\mathbf{v} = (\mathbf{W}^T)^{-1}(\mathbf{H}\mathbf{c} - \mathbf{g})$. The iterative procedure is as follows for a given λ:

1. Get the initial fits $\hat{f}_j^{\text{init}},\ j = 1, \ldots, p$ and $\hat{\boldsymbol{\beta}}_j^{\text{init}},\ j = 1, \ldots, q$.
2. Initialize $\hat{\mathbf{c}} = \mathbf{0}$.
3. Compute $\mathbf{g}$, $\mathbf{H}$, $\mathbf{W}$, and $\mathbf{v}$ based on the current value of $\hat{\mathbf{c}}$.
4. Minimize $\|\mathbf{v} - \mathbf{W}\mathbf{c}\|^2 + \lambda \left(\sum_{j=1}^{p} c_j + \sum_{j=1}^{q} d_j c_{p+j} \right)$, subject to $c_j \geq 0, j = 1, \ldots, p+q$ with the garrote routine for a given λ.

5. Repeat Steps 3 and 4 until the convergence criterion meets.

We use the garrote routine developed by Breiman (1995) in Step 4 of our algorithm.

Initial Estimation for Additive Cox Models

In this section, we propose a new approach to compute the initial estimates by fitting additive Cox models. We generalized the algorithm of Wood (2000, 2004), since it is desired to simultaneously estimate all components while incorporating automatic choices of smoothing parameters.

Define $\mathbf{u} = -\partial\ell_n/\partial\boldsymbol{\eta}$, $\mathbf{A} = -\partial^2\ell_n/\partial\boldsymbol{\eta}\boldsymbol{\eta}^T$ and $\boldsymbol{\gamma} = \boldsymbol{\eta} - \mathbf{A}^{-1}\mathbf{u}$, where $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_n)$. To reduce the computation burden, we replace $\mathbf{A}$ with a diagonal matrix $\mathbf{D}$ that has the same diagonal elements of $\mathbf{A}$. Hastie and Tibshirani (1990) argued this approximation leads to the similar results as the original $\mathbf{A}$. Let $\mathbf{D} = \mathbf{W}\mathbf{W}^T$. Then the Taylor expansion of $-\ell_n$ has the form of $\|\mathbf{W}(\mathbf{z} - \boldsymbol{\eta})\|^2$.

In our algorithm, we use the Newton-Raphson method to minimize the penalized partial likelihood. Since we use a basis expansion to represent the nonparametric regression, the total dimension of the estimator is large, which makes it hard to find a good starting point. The update of current solutions can easily overshoot the minimum, leading to the divergence of the algorithm. To solve this problem, we suggest using a fraction of original update, i.e, let $\boldsymbol{\gamma} = \boldsymbol{\eta} - \tau\mathbf{A}^{-1}\mathbf{u}$, where $0 < \tau < 1$. We call this backtracking.

To get the initial fit for additive Cox model, we use the following procedure:

1. Initialize $\hat{\boldsymbol{\eta}} = \mathbf{0}$.

2. compute $\mathbf{u}$,$\mathbf{W}$ and $\boldsymbol{\gamma}$ based on the current value.

3. Use $\boldsymbol{\gamma}$ as the pseudo responses and $\mathbf{W}$ as the weights to fit (Gaussian) additive model with *gam* subroutine in *mgcv*. Update $\hat{\boldsymbol{\eta}}$.

4. Evaluate the negative loglikelihood function in Step 3. If the solution overshoots in Step 3, we do a backtracking to find new $\hat{\boldsymbol{\eta}}$.

5. Repeat Steps 2, 3 and 4 until the convergence criterion meets.

Tuning Parameters

To estimate the tuning parameter λ in the garrote, we use the BIC criterion:

$$\text{BIC}_\lambda = -2\ell_n(\hat{\mathbf{c}}) + \text{df}_\lambda \times \log n, \tag{3.10}$$

where $\hat{\mathbf{c}}$ is the garrote solution for a given λ, and df_λ is the degree of freedom. To define df_λ, we use the definition of Yuan and Lin (2006). For categorical terms, we have

$$\text{df}_\lambda = 2\sum_{j=p+1}^{p+q} I(\hat{c}_j > 0) + \sum_{j=p+1}^{p+q} \hat{c}_j(d_{j-p} - 2). \tag{3.11}$$

For smoothing terms, the degree of freedom of the garrote is:

$$\text{df}_\lambda = 2\sum_{j=1}^{p} I(\hat{c}_j > 0) + \sum_{j=1}^{p} \hat{c}_j(\text{edf}_j - 2), \tag{3.12}$$

where edf_j is the effective degree of freedom of the estimated jth nonparametric component.

3.3 Simulation Studies

3.3.1 Example 1

In this example, we adopt the same model as in Leng and Zhang (2006). In particular, we generate eight variables X_j, $j = 1, \ldots, 8$, from $N(0, 1)$, and the correlation between X_i and X_j is $\rho^{|i-j|}$. Two case are considered: $\rho = 0$ and $\rho = 0.5$. Each variable is truncated into $[-2, 2]$ and scaled to $[0, 1]$. We set the baseline hazard function to be 1.

The true η is

$$\eta = f_1(X_1) + f_2(X_4) + f_3(X_7),$$

where

$$f_1(s) = 3(3s-2)^2, f_2(s) = 4\cos(\frac{(3s-1.5)\pi}{5}), f_3(s) = I(s < 0.5).$$

Also, we transform X_8 into a categorical variable $I(X_8 > 0.6)$. The censoring time C is generated from exponential distribution with mean $V\exp(-\eta(x))$, where V

Table 3.1: Selection frequency and the average number of correct 0 and incorrect 0 in Example 1 ($\rho = 0$).

n	censoring	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	Correct	Incorrect
100	15%	100	8	14	81	8	8	90	6	4.56	0.29
	30%	100	9	14	74	12	8	82	5	4.52	0.44
	45%	100	9	11	61	11	10	74	4	4.55	0.65
200	15%	100	6	4	98	9	5	99	4	4.72	0.03
	30%	100	6	5	96	8	8	97	3	4.70	0.07
	45%	100	4	4	83	10	6	96	2	4.74	0.21

Table 3.2: Average ISE estimation in Example 1 ($\rho = 0$).

n	censoring	Oracle ISE	ISE
100	15%	0.42	0.60
	30%	0.41	0.69
	45%	0.43	0.80
200	15%	0.23	0.34
	30%	0.25	0.36
	45%	0.25	0.40

The range of the standard errors for average oracle ISE is 0.02 to 0.05, for ISE is 0.02 to 0.06

is a random variable from $\mathrm{U}(a, a+2)$. We choose a such that the censoring rates are $15\%, 30\%$ and 45% respectively.

We consider two sample sizes $n = 100, 200$. In each simulation, we generate 100 data sets. To measure the model estimation accuracy, we calculate the integrated square error $\mathrm{ISE} = E(\sum_{j=1}^{p}(f_j - \hat{f}_j)^2)$, which is estimated by a Monte Carlo integration with 2000 testing points.

In Tables 3.1, 3.2, 3.3 and 3.4, we report the oracle ISE, which is obtained by fitting an additive Cox model with only important variables, the ISE of the garrote and the selection frequency of variables that appear in the final model. We also summarize the number of zero variables selected correctly by the model space (denoted as 'Correct') and the number of nonzero variables incorrectly set to zero (denoted as 'Incorrect'). These tables show that our method performs well in both variable selection and

Table 3.3: Frequency of selected variables and the average number of correct 0 and incorrect 0 in Example 1 ($\rho = 0.5$).

n	censoring	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	Correct	Incorrect
100	15%	100	7	10	76	10	13	87	11	4.49	0.37
	30%	100	14	19	73	16	15	85	12	4.24	0.42
	45%	100	16	19	53	19	17	72	19	4.10	0.75
200	15%	100	8	10	99	17	9	99	2	4.54	0.02
	30%	100	14	10	95	14	8	97	1	4.53	0.08
	45%	100	5	17	88	10	8	93	1	4.59	0.19

Table 3.4: The average ISE results summary in Example 1 ($\rho = 0.5$).

n	censoring	Oracle ISE	ISE
100	15%	0.45	0.63
	30%	0.44	0.68
	45%	0.51	0.96
200	15%	0.26	0.35
	30%	0.26	0.38
	45%	0.28	0.42

The range of the standard errors for average oracle ISE is 0.02 to 0.06, for ISE is 0.02 to 0.09

estimation.

In Figure 3.1, we plot one typical initial estimate for each component function with our algorithm. We can see our approach can produce very good initial estimates.

In Figure 3.2, we plot the estimated functions selected by the garrote with performance at 10th, 50th and 90th percentiles of the estimated ISE among the 100 simulations. To find the sampling variability of the garrote estimates at each point, we plot the 2.5th, 50th and 97.5 percentiles of the estimated functions at 100 random data points among 100 simulations in Figure 3.3. This forms a 95% pointwise empirical confidence interval for the garrote estimates. These results show that our method can identify important variables and provide very accurate estimation for each component function.

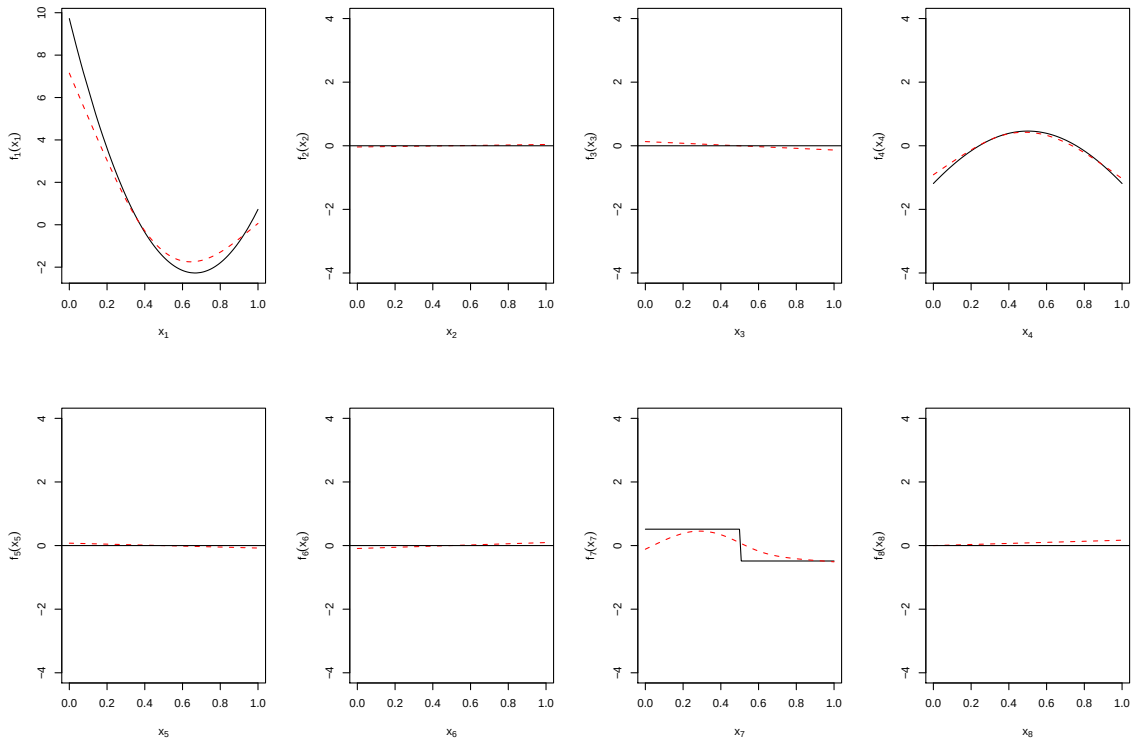

Figure 3.1: The initial estimates of component functions in additive Cox models.The solid lines indicate the true curves, the red dashed lines are initial estimates.

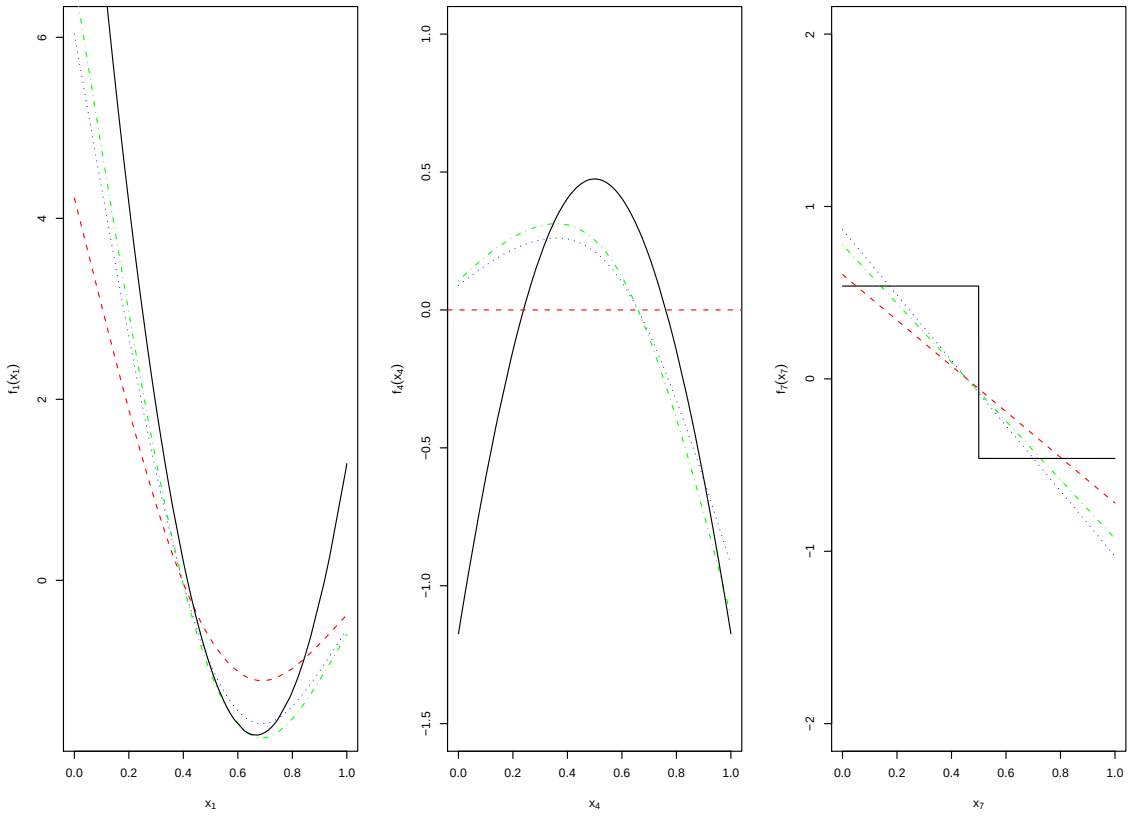

Figure 3.2: The garrote estimates of component functions when $n = 100$, $\rho = 0.5$ and the censoring rate is 45%. Red dashed lines indicate the 10th best, blue dotted lines indicate the 50th best, and green dot-dashed lines are the 90th best. The black solid lines are the true curves.

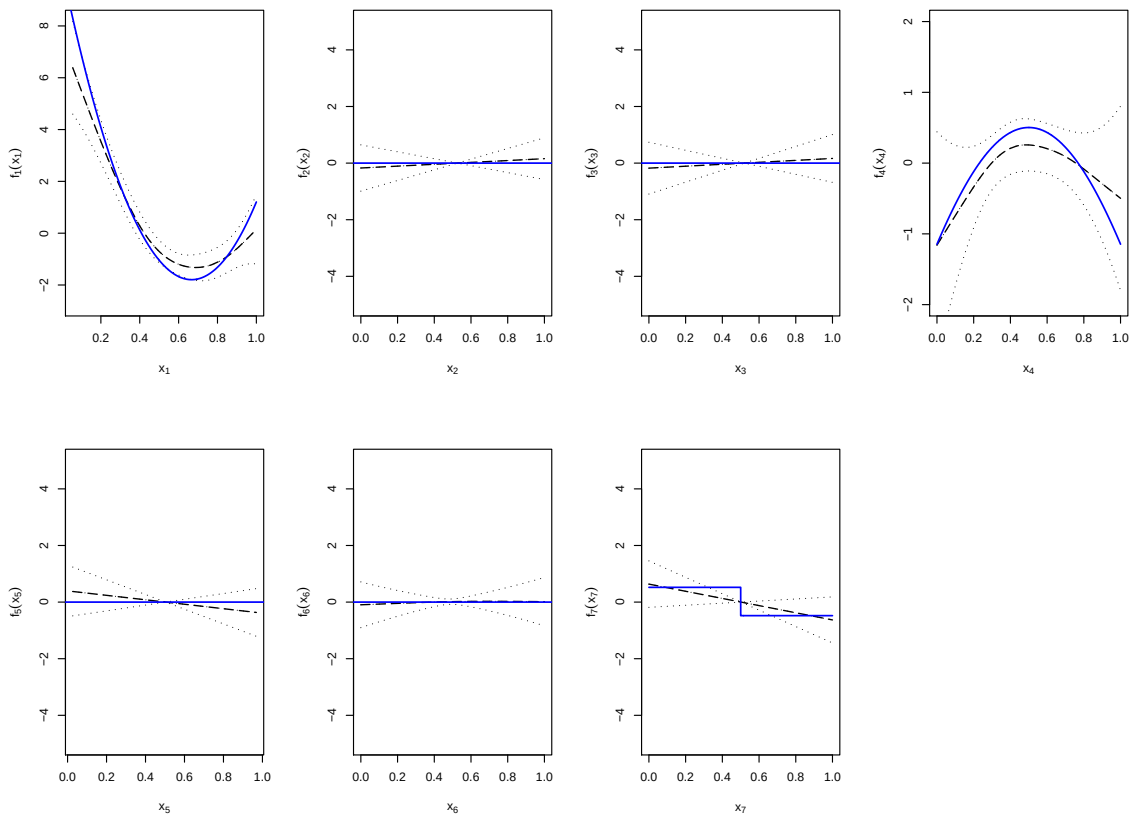

Figure 3.3: Sampling variability of the estimated functions with garrote at each point based on 100 simulations when $n = 100$, $\rho = 0.5$ and the censoring rate is 45%. The dotted lines are the 2.5th and 97.5th percentiles of 100 simulations, and they form the 95% envelopes. The long dash lines are the 50th percentiles. The blue solid lines are the true curves.

3.3.2 Example 2

We extend Example 1 into a high dimensional setting. In this simulation, we generate 20 variables X_j, $j = 1, \ldots, 20$ from $N(0, 1)$, and the pairwise correlation between X_i and X_j is $\rho^{|i-j|}$. Two cases are considered: $\rho = 0$ and $\rho = 0.5$. Each variable is truncated into $[-2, 2]$ and scaled to $[0, 1]$. We set the baseline hazard function to be 1.

The true η is

$$\eta = f_1(X_1) + f_2(X_2) + f_3(X_3),$$

where

$$f_1(s) = 3(3s-2)^2, f_2(s) = 4\cos(\frac{(3s-1.5)\pi}{5}), f_3(s) = I(s < 0.5).$$

Also, we transform X_j, $j = 18, 19, 20$, into categorical variables $I(X_j > 0.6)$. The censoring time C is generated from exponential distribution with mean $V\exp(-\eta(x))$, where V is a random variable from $\mathrm{U}(a, a+2)$. We choose a such that the censoring rates are $15\%, 30\%$ and 45% respectively.

The sample size $n = 300$. For each simulation, we generate 100 data sets. To measure the model estimation accuracy, we calculate the integrated square error $\mathrm{ISE} = E(\sum_{j=1}^{p}(f_j - \hat{f}_j)^2)$, which is estimated by a Monte Carlo integration with 2000 testing points.

Table 3.5: Selection frequency in Example 2 ($\rho = 0$).

censoring	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
15%	100	87	92	10	11	10	9	8	6	11
30%	100	77	85	13	14	11	10	12	9	13
45%	100	65	82	15	14	17	15	16	13	13
censoring	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}	X_{17}	X_{18}	X_{19}	X_{20}
15%	9	7	11	10	6	5	8	12	11	9
30%	11	12	15	11	9	9	13	16	12	11
45%	12	16	12	12	14	6	15	16	14	13

In Tables 3.5, 3.6 and 3.7, we report the oracle ISE, which is obtained by fitting an additive Cox model only important variables, the ISE of the garrote and the

Table 3.6: Selection frequency in Example 2 ($\rho = 0.5$).

censoring	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
15%	100	82	76	13	12	16	9	11	13	8
30%	99	79	71	16	17	15	18	12	14	12
45%	96	64	59	21	19	17	25	17	16	21
censoring	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}	X_{17}	X_{18}	X_{19}	X_{20}
15%	13	14	11	15	14	15	10	12	13	11
30%	11	15	10	13	15	17	13	15	15	17
45%	15	17	11	13	11	19	15	21	31	26

Table 3.7: The average ISE results summary in Example 2.

ρ	censoring	Oracle ISE	ISE
0	15%	0.21	0.72
	30%	0.23	0.73
	45%	0.25	0.81
0.5	15%	0.24	0.81
	30%	0.25	0.92
	45%	0.27	1.01

The range of the standard errors for average oracle ISE is 0.01 to 0.02, for ISE is 0.01 to 0.07

selection frequency of variables that appear in the final model. These results show that our initial fitting method is computationally stable in large p, large n settings. Our garrote procedure also performs well for both variable selection and estimation, even when the censoring rate is as high as 45%.

3.4 Primary Biliary Cirrhosis (pbc) Data

The pbc data was gathered from the Mayo Clinic trial in primary biliary cirrhosis of liver conducted between 1974 and 1984. This data is provided by Therneau and Grambsch (2000). In this study, 312 patients from a total of 424 patients who agreed to participate in the randomized trial are eligible for the analysis. For each patient, clinical and other related characteristics are collected. Of those, 125 patients died

before the end of follow-up. We use the 17 covariates as followings: age (in days), alb (albumin ib g/dl), alk (alkaline phosphatase in U/liter), bil (serum bilirubin in mg/dl), chol (serum cholesterol in mg/dl), cop(urine copper in μg/day), plat (platelets per cubic ml/1000), prot (prothrombin time in seconds), sgot (liver enzyme in U/ml), trig (triglycerides in mg/dl), asc (0, absence of ascites; 1, presence of ascites), ede (0, no edema; 0.5, untreated or successfully treated; 1, unsuccessfully treated edema), hep (0, absence of hepatomegaly; 1, presence of hepatomegaly), sex (0, male; 1, female), spid (0, absence of spiders; 1, presence of spiders), stage (histological stage of disease, grades $1, 2, 3$ or 4), trt (1, control; 2, treatment). The first ten variables are continues, the last seven are categorical.

We only use 276 complete observations. Among the total 17 variables, 10 of them are continues, and the rest are categorical. We compare our method with lasso, adaptive lasso (Alasso) and Cosso-type method proposed by Leng and Zhang (2007). In Table 3.8, we list the variables selected by each method (denoted by 1). Our method results in a simplest model by selecting 6 variables: 5 continuous variables and one categorical variable. These variables are also selected by other three methods.

The fitted continuous functions of our garrote model are plotted in Figure 3.4. Leng and Zhang (2007) concluded that most of the selected continuous functions selected by their method are linear. Our procedure shows that the fitted components all have nonlinear trends, which are worth further investigation.

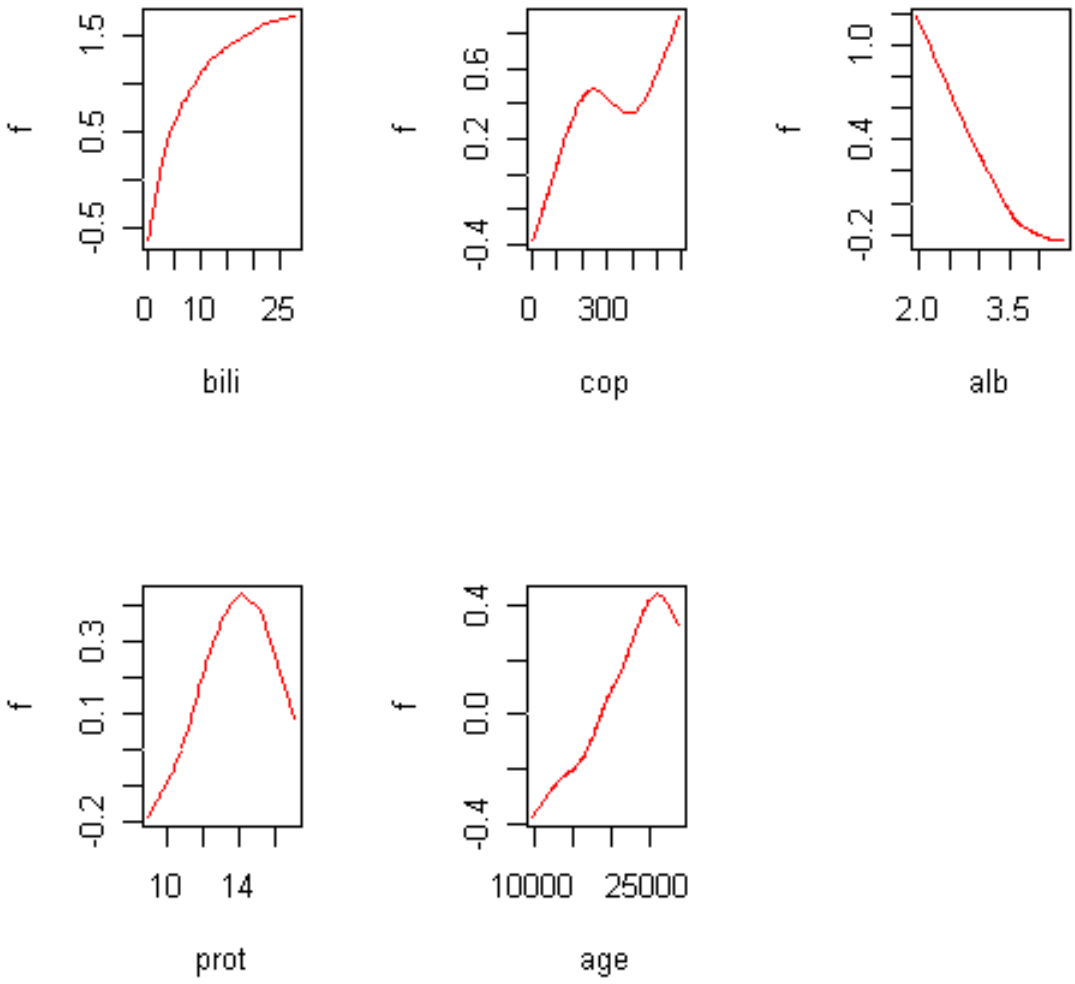

Figure 3.4: Garrote estimate for component functions in pbc data.

Table 3.8: Selected variables for pbc data.

Covariates	lasso	Alasso	Cosso-type	garrote
trt	0	0	0	0
age	1	1	1	1
sex	0	0	1	0
asc	1	0	0	0
hep	0	0	0	0
spid	0	0	0	0
ede	1	1	1	1
bil	1	1	1	1
chol	0	0	1	0
alb	1	1	1	1
cop	1	1	1	1
alk	0	0	0	0
sgot	1	1	1	0
trig	0	0	0	0
plat	0	0	0	0
prot	1	1	1	1
stage	1	1	1	0

Chapter 4

New Variable Selection Method in High Dimensional Data

4.1 Motivation

4.1.1 High Dimensional Data Challenge

There are more and more high dimensional data in real life. For example, high dimensional expression microarray data have been rapidly accumulated in the field of biological and medical research. They impose new challenges to conventional statistical model estimation and variable selection methods, mainly due to their unique "large p, small n" data structure. That is, these data in general contain large numbers of variables, typically tens of thousands of genes, but much smaller numbers of samples which are often less than hundreds. Effective and reliable variable selection methods are hence demanded to discover hidden patterns in these high dimensional data.

In the modern age, new technologies of data collection rapidly produce data sets of ever increasing samples sizes and dimensions (the number of variables), which often include superfluous variables. In order to enhance the generalization performance of a learning algorithm, it is important to identify important variables and build parsimonious classifiers with high interpretability and improved prediction performance.

In the research of cancer classification using microarray gene expression data, accurate identification of different cancer types is critical to achieve effective treatment and longer survival time of patients. Each type of cancer may be characterized by a group of abnormally expressed genes, called a "signature". Since gene expression arrays measure tens of thousands of genes, choosing the ones that comprise a signature that can accurately classify cancer subtypes remains a major challenge. It is therefore desired to select important features, which help to build highly interpretable classifiers with competitive generalization performance.

4.1.2 Two-stage Ranking Method

For gene selection, commonly used methods include qualitative observations, heuristic rules such as cutoff values and model-based probability analysis. Various two-sample t-tests procedures have been proposed including both parametric tests and nonparametric tests. Bayesian approaches generally assume priors which favor sparseness of the model.

Variable selection by first ranking variables based on the association strength of individual variables to the response variable and then selecting the top ones is another popular technique used in practice, especially for high-dimensional linear models. When the data dimension is high with $p > n$ or ultra-high with $p \gg n$, some selection methods such as backward elimination can not be directly applied since the OLS estimate can not be uniquely defined. One natural way to handle this is a two-stage procedure: pre-screening followed by model fitting. We can first screen data by eliminating redundant features and reducing the data dimension from p to p^* such that $p^* < n$, and then apply a standard selection procedure to the reduced data. For example, in cancer classification using microarray gene expression data, gene-ranking is often used to identify "marker" genes which characterize different types of cancer.

The screening step is often implemented by ordering variables based on some measures of their "importance", which needs to be carefully chosen to assure that no important variables would be filtered out. A good ranking criteria is very important for variable screening, in the sense that its magnitude should well reflect the strength

of the association between individual variables and the response variable, such that the variables that have strong correlation with the response can be more likely retained in the model than those having weak correlation with the response. To overcome the curse of dimensionality, most ranking methods are based on a univariate model fitting and inference for each input variable. In linear models, each predictor is assumed to be linearly related with the response, therefore marginal correlation coefficients are often used to order them. Various correlation coefficients have been proposed as ranking criteria, such as Fisher correlation coefficient, t-statistic, p-value, BW ratio (Between-class Within-class variation ratio), and etc.

Variable ranking based on marginal correlation is very useful for dimension reduction, but it may not work well in the case of collinearity and nonlinearity. For high dimensional data, many variables tend to be highly correlated with each other. Sometimes one unimportant predictor can be highly correlated with important predictors, and is therefore more likely to be selected than other important predictors which have weak marginal correlation with the response. Also, the marginal correlation ranking ignores the nonlinearity effects between the dependent variable and predictors.

4.2 New Variable Selection Method in High Dimensional Data

For high dimensional data with $p > n$, most of current nonparametric methods can not handle this due to the small number of observations. L_2 boosting with componentwise smoothing splines (Buhlmann and Yu, 2003) can conduct variable selection and estimation simultaneously. But in cases of $p \gg n$, the componentwise boosting tends to overfit (Buhlmann, 2008) and the number of selected variables is still large. In the following, we propose a two-stage boosting algorithm: at the first stage, we screen all the variables by fitting the GAM by boosting componentwise regression and smoothing; at the second stage, we use our garrote method to select variables from the variables preselected by first stage boosting.

4.2.1 Automatic Screening by Boosting

In this step, we use boosting as a screening tool to exclude some unimportant variables and hence achieve dimension reduction. We use the penalized thin-plate regression splines as weak learners. The loss function is the negative likelihood (deviance). Define $\mathbf{y} = (y_1, \ldots, y_n)^T$, $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_q)^T$, $\mathbf{f}_j = (f_j(x_{j1}), \ldots, f_j(x_{jn}))^T$, $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_q)$, $\mathbf{1}_n = (1, \ldots, 1)^T$, and $\mathbf{0}_n = (0, \ldots, 0)^T$. The boosting procedure is as following:

1. Initialization.

$$\begin{aligned}
\hat{\boldsymbol{\alpha}}^{(0)} &= \bar{y}\mathbf{1}_n, \\
\hat{\boldsymbol{\beta}}^{(0)} &= (\mathbf{0}_{d_1}, \ldots, \mathbf{0}_{d_q})^T, \\
\hat{\mathbf{f}}_j^{(0)} &= \mathbf{0}_n, j = 1, \ldots, p, \\
\hat{\boldsymbol{\eta}}^{(0)} &= \bar{y}\mathbf{1}_n, \\
\hat{\boldsymbol{\mu}}^{(0)} &= g^{-1}(\hat{\boldsymbol{\eta}}^{(0)}).
\end{aligned}$$

2. For $m = 0, 1, \ldots, M$ do;

- **Fit parametric (linear) terms**

  For $j = 1, \ldots, q$, do;

  - Estimate $\hat{\alpha}_j$ and $\hat{\boldsymbol{\beta}}_j$ with $(\mathbf{y} - \hat{\boldsymbol{\mu}}^{(m)})$ as response, using GLM in R.
  - Let $\hat{\boldsymbol{\eta}}_{j,\text{new}} = \hat{\boldsymbol{\eta}}^{(m)} + \hat{\alpha}_j + \mathbf{z}_j\hat{\boldsymbol{\beta}}_j$.

  end;

  Choose $\mathbf{z}_\iota$, such that $\iota = \max_{j=1,\ldots,q} \Big(\text{Deviance}(\hat{\boldsymbol{\eta}}^{(m)}) - \text{Deviance}(\hat{\boldsymbol{\eta}}_{j,\text{new}})\Big)$.

  Update:

$$\begin{aligned}
\hat{\boldsymbol{\alpha}}^{(m+1)} &= \hat{\boldsymbol{\alpha}}^{(m)} + \nu\hat{\alpha}_\iota\mathbf{1}_n, \\
\hat{\boldsymbol{\beta}}_\iota^{(m+1)} &= \hat{\boldsymbol{\beta}}_\iota^{(m)} + \nu\hat{\boldsymbol{\beta}}_\iota, \\
\hat{\boldsymbol{\beta}}_j^{(m+1)} &= \hat{\boldsymbol{\beta}}_j^{(m)},\ j = 1, \ldots, q,\ \textit{and}\ j \neq \iota, \\
\hat{\boldsymbol{\eta}}_{\text{temp}} &= \hat{\boldsymbol{\alpha}}^{(m+1)} + \mathbf{Z}\hat{\boldsymbol{\beta}}^{(m+1)} + \sum_{j=1}^{p} \hat{\mathbf{f}}_j^{(m)}.
\end{aligned}$$

- **Fit smoothing terms**

  For $j = 1, \ldots, p$, do;

  - Estimate $\hat{\mathbf{f}}_{j,\text{temp}}$ with $(\mathbf{y} - g^{-1}(\hat{\boldsymbol{\eta}}_{\text{temp}}))$ as response, using $gam(mgcv)$ in R, and setting all the smoothing parameters to be large.
  - Let $\hat{\boldsymbol{\eta}}_{j,\text{temp}} = \hat{\boldsymbol{\eta}}_{\text{temp}} + \hat{\mathbf{f}}_{j,\text{temp}}$.

  end;

  Choose the κth term,

  where $\kappa = \max_{j=1,\ldots,p} \big(\text{Deviance}(\hat{\boldsymbol{\eta}}_{\text{temp}}) - \text{Deviance}(\hat{\boldsymbol{\eta}}_{j,\text{temp}})\big)$.

  Update:

$$\begin{aligned} \hat{\mathbf{f}}^{(m+1)} &= \hat{\mathbf{f}}^{(m)} + \hat{\mathbf{f}}_{\kappa,\text{temp}}, \\ \hat{\boldsymbol{\eta}}^{(m+1)} &= \hat{\boldsymbol{\eta}}_{\text{temp}} + \hat{\mathbf{f}}^{(m+1)}, \\ \hat{\boldsymbol{\mu}}^{(m+1)} &= g^{-1}(\hat{\boldsymbol{\eta}}^{(m+1)}) \end{aligned}$$

  Update m to $m + 1$.

Stop the boosting step.

In practice, we set $M = 500$, $\nu = 0.1$. We choose the "optimal" boosting step at which the AIC achieves minimum. For AIC, we need to calculate the hat matrix, which can be derived using the method in Tutz and Binder (2006). After boosting screening, we selected p^* continues variables and q^* categorical variables. We often observed that $p^* + q^* < n$. So after the first-stage screening, the traditional methods can be applied in the second stage.

4.2.2 Second-stage Garrote

For the pre-selected variables retained from the first stage, we further select the important variables by taking the following procedure:

1. Get the initial fits from the first-stage boosting: $\hat{f}_j^{\text{init}}$, $j = 1, \ldots, p^*$ and $\hat{\boldsymbol{\beta}}_j^{\text{init}}$, $j = 1, \ldots, q^*$.

2. Initialize $\hat{\mathbf{c}} = \mathbf{0}$.

3. Compute $\mathbf{g}$, $\mathbf{H}$, $\mathbf{W}$, and $\mathbf{v}$ based on the current value of $\hat{\mathbf{c}}$.

4. Minimize $\|\mathbf{v} - \mathbf{W}\mathbf{c}\|^2 + \lambda \left(\sum_{j=1}^{p^*} c_j + \sum_{j=1}^{q^*} d_j c_{p^*+j} \right)$, subject to $c_j \geq 0, j = 1, \ldots, p^* + q^*$ with the garrote routine for given λ.

5. Repeat Steps 3 and 4 until the convergence criterion meets.

This method can also be extended into survival data, using partial likelihood to replace the deviance in our algorithm, and then we can conduct variable selection and model estimation for high dimensional survival data.

4.3 Simulation Studies

Consider a situation where the dimension of predictors is higher than the sample size (large p, small n). We generate 500 independent variables X_j, $j = 1, \ldots, 500$, and only the first three variables are relevant. The true logit function is:

$$\eta(\mathbf{X}) = f_1(X_1) + f_2(X_2) + f_3(X_3), \tag{4.1}$$

where

$$f_1(s) = 5s;\ f_2(s) = 3(2s-1)^2;\ f_3(s) = \frac{4\sin(2\pi s)}{2 - \sin(2\pi s)}. \tag{4.2}$$

Sample sizes are $40, 50, 60, 70, 100$ and y_i is generated independently from $\text{Bin}(1, \exp(\eta_i)/(1+\exp(\eta_i)))$. For each simulation, we generate 100 data sets. To measure the model estimation accuracy, we calculate the integrated square error $\text{ISE} = E(\sum_{j=1}^{p}(f_j - \hat{f}_j)^2)$. It is estimated by a Monte Carlo integration with 10000 testing points. We also calculate misclassification errors (denoted as "Error") for each simulation. In each Monte Carlo simulation, we report the oracle ISE and oracle misclassification error, which are obtained by fitting the logistic additive model by including important variables only. We also report the model size which is the number of selected variables in the final model. The Bayes error is 0.081.

Tables 4.1 shows that the average model size of two-stage garrote increases with the increasing sample size. The average ISE approaches to oracle ISE when the sample

Table 4.1: Average ISE , misclassification errors and model sizes.

n	Model Size	ISE	Oracle ISE	Error	Oracle error
40	5.17	42	16	0.10	0.09
50	6.20	21	16	0.10	0.09
60	7.07	16	14	0.10	0.09
70	7.67	16	13	0.09	0.09
100	11.52	14	13	0.09	0.09

The range of the standard errors for average model size is 0.20 to 0.36, for average ISE is 1 to 4, for average oracle ISE is 1 to 2, for average error is 0.00, for oracle error is 0.00.

size increases. It is observed that the misclassification error is relatively small, even in very small sample.

4.4 UNC Breast Cancer Data

Three public microarray gene expression datasets, "Stanford", "Rosetta" and "Singapore", are combined and used in this study. The details of these three datasets are described in Zhang, *et al.* (2005). The combined data set has $p = 2924$ genes and $n = 300$ patients. Our purpose is to find the most important variables and use them to classify the tissues into: cancer or non-cancer. It is a typical "large p, small n" classification problem. Support vector machine (SVM) is a standard classification method to deal with it. But it can suffer from the existence of redundant variables (Hastie *et al.*, 2001). Bradly and Mangasarian (1998) proposed L_1 SVM which by applying the lasso type penalty on the hyperplane coefficients. Zhang *et al* (2005) introduced SCAD SVM, and they imposed a nonconvex penalty which is used in SCAD on the coefficients. We apply our two-stage garrote method to this data, and compare it with SVM, L_1 SVM and SCAD SVM.

We follow the same procedure as Zhang *et al.* (2005) used in their study to do model training and testing. We use the source information to separate the combined data into three folds naturally: "Stanford", "Rosetta" and "Singapore". The sample size for each fold is: 104, 97 and 99 respectively. We train each model on two folds

and test it on the third one. For example, we first train our garrote on "Rosetta" and "Singapore", then we test it and calculate misclassification errors on the third one, "Stanford". We call it Stanford learning. We repeat this procedure in a similar way and got Rosetta learning and Singapore learning. For the two-stage garrote, we use AIC to decide when the first-stage boosting produces an "optimal" set of screened variables. In Figure 4.1, we plot AIC for varying number of boosting steps in Stanford learning. In this learning, we use the selected variables at step 108 at which AIC reaches minimum.

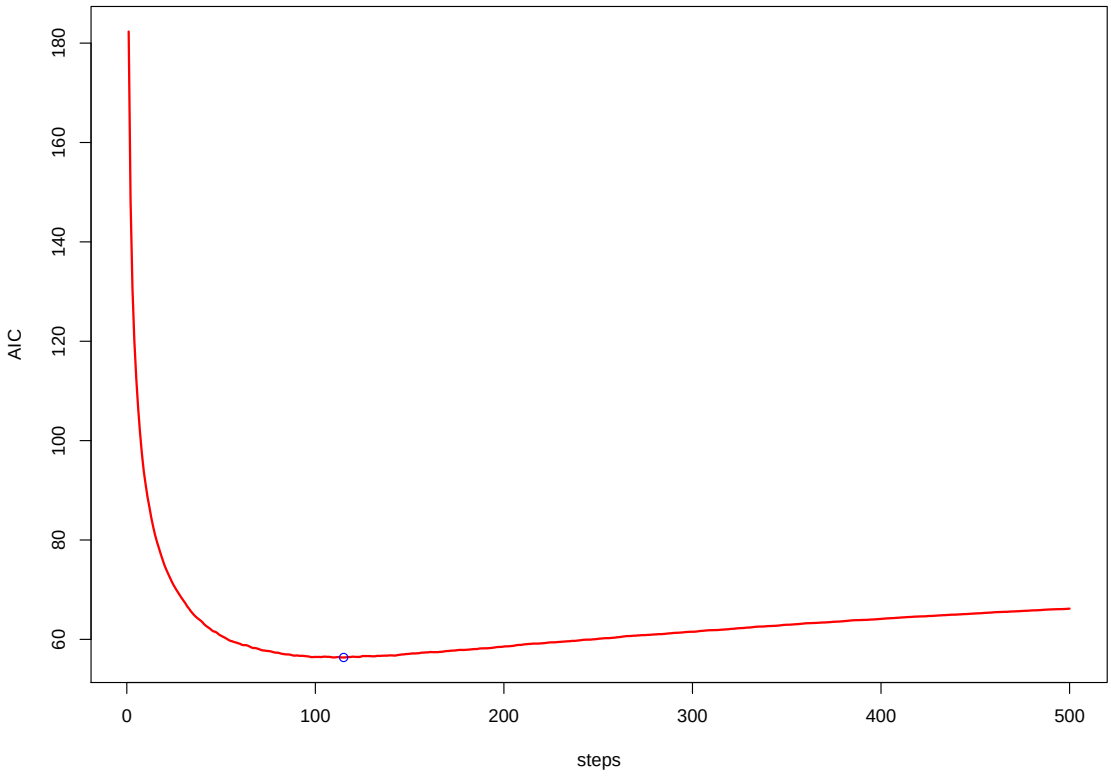

Figure 4.1: AIC versus the number of boosting steps for UNC data. Blue circle indicates the minimum of AIC

Table 4.2 shows the test misclassification error in each learning and the average error for the SVM, L_1 SVM, SCAD SVM. As a reference, we also include the result

Table 4.2: Three-fold cross validation errors for UNC data.

	Stanford	Rosetta	Singapore	Average
SVM	0.154	0.175	0.051	0.127
L_1 SVM	0.125	0.216	0.081	0.141
SCAD SVM	0.115	0.175	0.061	0.117
first-stage boosting	0.126	0.221	0.068	0.138
two-stage garrote	0.106	0.165	0.060	0.110

from our first-stage boosting fitting. We denote it as "first-stage boosting". In both Stanford and Rosetta learning, the two-stage garrote has the smallest errors. In Singapore learning, our method is the second best, only next to SVM. The first-stage boosting has a smaller error than SVM in Stanford learning, and a smaller error than L_1 SVM in Singapore learning, although it has largest errors in Rosetta learning. Overall, the two-stage garrote has the smallest error among all the methods under comparison.

Table 4.3: Number of selected genes for UNC data.

	Stanford	Rosetta	Singapore	Average
L_1 SVM	59	63	72	65
SCAD SVM	15	19	31	22
first-stage boosting	32	22	28	27
two-stage garrote	16	14	16	16

In Table 4.3, we show the number of genes selected in each learning. Since SVM uses all the variables, we only compare the two-stage garrote, the first-stage boosting, the L_1 SVM and the SCAD SVM. The L_1 SVM selects 65 genes on average, but the SCAD SVM and our methods select much less genes than the L_1 SVM. Our two-stage garrote selects the fewest genes and has the smallest misclassification error. In Table 4.4, we summarize the UniGene identifiers of all the genes in three learnings that are selected at least twice by our two-stage garrote (garrote). The third and fourth columns are respectively the frequency of each gene being selected by the SCAD SVM (SCAD) and L_1 SVM (L_1). The last column displays the corresponding annotation of UniGene identifiers. We can see the top genes selected by our methods

are also selected by other methods frequently. Hs169946 is selected three times by all of the methods, indicating that it might be a key gene for signal transduction in the biological pathway.

Table 4.4: Gene selection frequency for UNC data.

UGid	garrote	SCAD	L_1	gene annotation
Hs169946	3	3	3	GATA binding protein 3
Hs80420	3	3	2	Chemokine (C-X3-C motif) ligand 1
Hs298654	3	2	2	Dual specificity phosphatase 6
Hs79136	2	3	2	Solute carrier family 39
Hs1657	2	2	3	estrogen receptor 1
Hs26770	2	2	3	fatty acid binding protein 7, brain
Hs191842	2	0	3	cadherin 3, type 1, P-cadherin (placental)

Chapter 5

Discussion

In this dissertation study, we propose a new method for variable selection in semi-parametric additive models. We investigate the procedures in different model settings: generalized additive models and additive Cox models. We also extend our procedure to generalized additive models to high dimensional data. Our garrote method can make variable selection and estimation for both nonparametric functions and linear predictors. With our garrote, continuous variables and categorical variables are treated in a unified formulation.

For generalized additive models, we develop an efficient algorithm to solve the garrote based on the likelihood. We emphasize that the success of our method of variable selection depends on quality of initial estimates. A stable algorithm for estimating initial solutions based on boosting is developed when the sample size is relatively small compared to the dimensionality of predictors. The proposed algorithm produces a fairly good initial fit for component functions. We compare our method with other procedures, and the simulation results show that our garrote produce a sparser but highly predictable models than other methods. When there is strong correlation among the predictors, the garrote tends to drop important variables more often, but still leads to a model with good prediction. The real examples suggest that the garrote generally outperforms other competitive methods in terms of prediction.

High dimensional data are becoming more common in many areas, and they impose serious challenges to statistical estimation and model selection. We further

extend our procedure to high dimensional data, in which the samples size is much smaller than the number of model predictors. We propose a two-stage method to conduct variable selection. The simulation shows that our method can produce a model with a reasonably small size and good prediction. A real example of microarray data analysis suggests the proposed two-stage garrote results good models with low prediction errors and high sparsity, which are exactly sought in microarray studies.

We also generalize our method to survival data by focusing on the semi-parametric Cox models. Since there is little research on the initial estimation for additive Cox model, we develop an efficient algorithm to get the initial component estimates in semi-parametric Cox models. We then propose a new variable selection method based on the partial likelihood. Simulations show that our procedure can choose variables accurately, even in highly censored data. Also, the selected components can be estimated very well.

Future research will include extending the proposed procedures to other semi-parametric regression models such as zero-inflation models. Oracle properties for these semi-parametric models may exist, which are worth further investigation.

Bibliography

Akaike, H. (1973). Maximum likelihood identification of Gaussian autoregressive moving average models. *Biometrika* **60**, 255–265.

Blake, C. L. & Merz, C. J. (1998). 'UCI machine learning repository'.

Breiman, L. (1995). Better subset regression using the nonnegative garrote. *Technometrics* **37**, 373–384.

Breiman, L. (1998). Arcing classifier (Pkg: P801-849). *The Annals of Statistics* **26**(3), 801–824.

Breiman, L., Friedman, J., Olshen, R. & Stone, C. (1984). *Classification and Regression Trees*. Wadsworth and Brooks. Monterey, CA.

Bühlmann, P. & Yu, B. (2005). Boosting, model selection, lasso and nonnegative garrote. Seminar 127. Für Statistik ETH Zürich.

Cantoni, E., Flemming, J. M. & Ronchetti, E. (2006). Variable selection in additive models by nonnegative garrote. Cahiers du Dpartement d'Economtrie 2006.02. Dpartement d'Economtrie, Universit de Genve.

Efron, B., Hastie, T., Johnstone, I. & Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics* **32**(2), 407–499.

Fahrmeir, L. & Lang, S. (2001). Bayesian inference for generalized additive mixed models based on Markov random field priors. *Journal of the Royal Statistical Society, Series C: Applied Statistics* **50**(2), 201–220.

Fahrmeir, L., Kneib, T. & Lang, S. (2004). Penalized structured additive regression for space-time data: A Bayesian perspective. *Statistica Sinica* **14**(3), 731–761.

Fan, J. & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association* **96**(456), 1348–1360.

Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Inf. Comput.* **121**(2), 256–285.

Freund, Y. & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *in* 'International Conference on Machine Learning'. pp. 148–156.

Friedman, J. H. (1991). Multivariate adaptive regression splines (Disc: P67-141). *The Annals of Statistics* **19**, 1–67.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine.. *The Annals of Statistics* **29**(5), 1189–1232.

Friedman, J. H. & Silverman, B. W. (1989). Flexible parsimonious smoothing and additive modeling (with discussion). *Technometrics* **31**, 3–39.

Friedman, J., Hastie, T. & Tibshirani, R. (2000). Additive Logistic Regression: a Statistical View of Boosting. *The Annals of Statistics*.

Fu, W. (1998). Penalized regressions: the bridge versus the lasso. *Journal of Computational and Graphical Statistics* **7**, 397–416.

Gu, C. (2002). *Smoothing Splines ANOVA Models*. Spinger-Verlag. New York.

Hastie, T. (1989). Discussion of "Flexible parsimonious smoothing and additive modeling" by J. Friedman and B. Silverman. *Technometrics* **31**, 3–39.

Hastie, T., Tibshirani, R. & Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag Inc.

Leng, C. & Zhang, H. H. (2006). Model selection in nonparametric hazard regression. *Journal of Nonparametric Statistics* **18**(7-8), 417–429.

Lin, Y. & Zhang, H. H. (2006). Component selection and smoothing in smoothing spline analysis of variance models – COSSO. *Annals of Statistics* **5**, 2272–2297.

Mallows, C. L. (1973). Some comments on C_p. *Technometrics* **15**, 661–675.

Mangasarian, O. & Musicant, D. (1999). 'Massive support vector regression'.

Monahan, J. F. (2001). *Numerical Methods of Statistics*. Cambridge University Press.

Picard, R. R. & Cook, R. D. (1984). Cross-validation of regression models. *Journal of the American Statistical Association* **79**, 575–583.

Rätsch, G., Onoda, T. & Müller, K.-R. (2001). Soft margins for AdaBoost. *Machine Learning* **42**(3), 287–320. also NeuroCOLT Technical Report NC-TR-1998-021.

Schapire, R. E. (1990). The strength of weak learnability. *Mach. Learn.* **5**(2), 197–227.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics* **6**, 461–464.

Therneau, T. M. & Grambsch, P. M. (2000). *Modeling Survival Data: Extending the Cox Model*. Springer-Verlag.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B: Methodological* **58**, 267–288.

Tibshirani, R. (1997). The lasso method for variable selection in the Cox model. *Statistics in Medicine* **16**, 385–395.

Tutz, G. & Binder, H. (2006). Generalized additive modeling with implicit variable selection by likelihood-based boosting. *Biometrics* **62**(4), 961–971.

Wahba, G. (1990). Spline models for observation data. Vol. 59 of *CBNS-NSF Regional Conference Series in Applied Mathematics*. SIAM.

Wood, S. N. (2000). Modeling and smoothing parameter estimation with multiple quadratic penalties. *Journal Of The Royal Statistical Society Series B* **62**(2), 413–428.

Wood, S. N. (2003). Thin plate regression splines. *Journal of the Royal Statistical Society, Series B: Statistical Methodology* **65**(1), 95–114.

Wood, S. N. (2004). Stable and efficient multiple smoothing parameter estimation for generalized additive models. *Journal of the American Statistical Association* **99**, 673–686.

Yuan, M. & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B* **68**(1), 49–67.

Yuan, M. & Lin, Y. (2007). On the non-negative garrotte estimator. *Journal Of The Royal Statistical Society Series B* **69**(2), 143–161.

Zhang, H. H. & Lin, Y. (2006). Component selection and smoothing for nonparametric regression in exponential families. *Statistica Sinica* **16**(3), 1021–1041.

Zhang, H. H., Ahn, J., Lin, X. & Park, C. (2006). Gene selection using support vector machines with non-convex penalty. *Bioinformatics* **22**(1), 88–95.

Zou, H. (2006). The adaptive LASSO and its oracle properties. *Journal of the American Statistical Association* **101**(476), 1418–1429.

Zou, H. & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B: Statistical Methodology* **67**(2), 301–320.

Appendix A

Variable Annotations

Table A.1: Variable annotations for German credit data (1).

Variable Description	Variable Type	Code Description
Checking account status	Categorical	A10 : < 0
		A11: $0 \leq \ldots < 200$
		A12 : ≥ 200
		A13: no checking account
Duration of credit	Numerical	
Credit history	Categorical	A30: no credits taken
		A31: all credits paid back duly
		A32: existing credits paid back
		A33: delay in paying off
		A34: critical account
Purpose	Categorical	A40 : car (new)
		A41 : car (used)
		A42 : furniture/equipment
		A43 : radio/television
		A44 : domestic appliances
		A45 : repairs
		A46 : education
		A47: vacation
		A48 : retraining
		A49 : business
		A410 : others
Credit amount	Numerical	
Savings account/bonds	Categorical	A61 : $\ldots < 100$
		A62 : $100 \leq \ldots < 500$
		A63 : $500 \leq \ldots < 1000$
		A64 : ≥ 1000
		A65 : unknown
employment	Categorical	A71 : unemployed
		A72 : < 1 year
		A73 : $1 \leq \ldots < 4$ years
		A74 : $4 \leq \ldots < 7$ years
		A75 : ≥ 7 years
Installment rate	numerical	

Table A.2: Variable annotations for German credit data (2).

Variable Description	Variable Type	Code Description
Personal status	Categorical	A91 : male : divorced/separated
		A92 : female : divorced/separated
		A93 : male : single
		A94 : male : married/widowed
		A95 : female : single
Other debtors	Categorical	A101 : none
		A102 : co-applicant
		A103 : guarantor
Residence	Numerical	
Property	Categorical	A121 : real estate
		A122 :life insurance
		A123 : car or other
		A124 : unknown / no property
Age	Numerical	
Other installment	Categorical	A141 : bank
		A142 : stores
		A143 : none
Housing	Categorical	A151 : rent
		A152 : own
		A153 : for free
Existing credits	Numerical	
Job	Categorical	A171 : unemployed
		A172 : unskilled
		A173 : skilled
		A174 : highly qualified employee
People being liable	Numerical	
Telephone	Categorical	A191: yes
		A192: no
Foreign worker	Categorical	A201 : yes
		A202 : no

Table A.3: Variable annotations for wpbc data.

Variable Name	Variable Description
status	a factor with levels N (nonrecur) and R (recur)
mean-radius	radius (mean of distances from center to perimeter)
mean-texture	texture (standard deviation of gray-scale values)
mean-perimeter	perimeter
mean-area	area
mean-smoothness	smoothness (local variation in radius lengths)
mean-compactness	compactness
mean-concavity	concavity (severity of concave portions of the contour)
mean-concavepoints	concave points (number of concave portions of the contour)
mean-symmetry	symmetry
mean-fractaldim	fractal dimension
SE-radius	radius (SE)
SE-texture	texture (SE)
SE-perimeter	perimeter (SE)
SE-area	area (SE)
SE-smoothness	smoothness (SE)
SE-compactness	compactness (SE)
SE-concavity	concavity (SE)
SE-concavepoints	concave points (SE)
SE-symmetry	symmetry (SE)
SE-fractaldim	fractal dimension (SE)
worst-radius	radius (worst)
worst-texture	texture (worst)
worst-perimeter	perimeter (worst)
worst-area	area (worst)
worst-smoothness	smoothness ((worst)
worst-compactness	compactness (worst)
worst-concavity	concavity (worst)
worst-concavepoints	concave points (worst)
worst-symmetry	symmetry (worst)
worst-fractaldim	fractal dimension (worst)
tsize	diameter of the excised tumor in centimeters
pnodes	number of positive axillary lymph nodes