SurvBoost: An R Package for High-Dimensional Variable Selection in the Stratified Proportional Hazards Model via Gradient Boosting

Emily Morris  Kevin He  Yanming Li  Yi Li  Jian Kang
University of Michigan, Ann Arbor, MI 48109

March 22, 2018

Abstract

High-dimensional variable selection in the proportional hazards (PH) model has many successful applications in different areas. In practice, data may involve confounding variables that do not satisfy the PH assumption, in which case the stratified proportional hazards (SPH) model can be adopted to control the confounding effects by stratification of the confounding variable, without directly modeling the confounding effects. However, there is lack of computationally efficient statistical software for high-dimensional variable selection in the SPH model. In this work, an R package, SurvBoost, is developed to implement the gradient boosting algorithm for fitting the SPH model with high-dimensional covariate variables and other confounders. Extensive simulation studies demonstrate that in many scenarios SurvBoost can achieve a better selection accuracy and reduce computational time substantially compared to the existing R package that implements boosting algorithms without stratification. The proposed R package is also illustrated by an analysis of the gene expression data with survival outcome in The Cancer Genome Atlas (TCGA) study. In addition, a detailed hands-on tutorial for SurvBoost is provided.

1 Introduction

Variable selection for high-dimensional survival data has become increasingly important in a variety of research areas. One of the most popular methods is based on the proportional hazards (PH) model. Many penalized regression methods including adaptive lasso and elastic net have been proposed for the PH model [18, 17, 8]. Alternatively, boosting described by Buhlmann and Yu [5] has been adopted for variable selection in regression models and the PH model via gradient descent techniques. It can have a better variable selection accuracy compared with other methods in many scenarios. The R package mboost has been developed and become a powerful tool for variable selection and parameter estimation in complex parametric and nonparametric models via the boosting methods [11]. It has been widely used in many applications.

However, in many biomedical studies, the collected data may involve confounding variables that do not satisfy the PH assumption. For example, in cancer research you may argue that gender effects are not proportional, but we are more interested in selecting genes as the important risk factors for cancer survival. The PH assumption can reasonably be imposed on modeling the gene effects but not for gender effects. In this case the stratified proportional hazards (SPH) models are needed. In particular, the data are often grouped into multiple strata according to confounding variables. The SPH model adjusts those confounding effects by fitting the Cox regression with different baseline hazards for different strata, while still assuming that the covariate effects of interest are the same across different strata and satisfy the proportional hazard assumption.

The SPH model has a wide range of applications for survival analysis, but no computationally efficient statistical software are available for high-dimensional variable selection in the SPH model. To fill this gap, we develop an R package, SurvBoost, to implement the gradient boosting algorithm for fitting the SPH model with high-dimensional covariates with adjusting confounding variables. SurvBoost implements the gradient decent algorithm for fitting both PH and SPH model. The algorithm for the PH model has been used for the additive Cox model in mboost package which cannot fit the SPH model to perform variable selection. In our SurvBoost package, we optimize the implementations which can
reduce 30%–50% computational time. Additional options are available in the SurvBoost package to determine an appropriate stopping criteria for the algorithm. Another useful function assists in selecting stratification variables, which may improve model fitting results.

The rest of the paper is organized as follows: In Section 2, we will provide a brief overview of the gradient boosting method for the SPH model along with the algorithm stopping criteria. In Section 3, we show that SurvBoost can achieve a better selection accuracy and reduce computational time substantially compared with mboost. In Section 4, we provide a detailed hands-on tutorial for SurvBoost. In Section 5, we illustrate the proposed R package on an analysis of the gene expression data with survival outcome in The Cancer Genome Atlas (TCGA) study.

2 Methods

2.1 Stratified Proportional Hazards Model

The Cox proportional hazards model is effective for modeling survival outcomes in many applications. An important assumption underlying this model is a constant hazard ratio, meaning that the hazard for one individual is proportional to that of any other individual. This is a strong assumption for many applications. Thus, one useful adaptation to this model is relaxing the strict proportional hazards assumption; one approach is to allow the baseline hazard to differ by group across the observations. This is known as the stratified proportional hazards (SPH) model.

Suppose the dataset consists of n subjects. For $i = 1, \ldots, n$, denote by $T_i$ the observed time of event or censoring for subject $i$ and $\delta_i$ indicates whether or not an event occurred for subject $i$. Denote by $G$ the total number of strata and by $n_g$ the number of subjects in stratum $g$. Let $g_i$ be the strata indicator for subject $i$. Suppose there are $p$ potential covariate variables of our interest to select. For $j = 1, \ldots, p$, let $x_{ij}$ be the covariate $j$ for subject $i$. For stratum $g = 1, \ldots, G$, the hazard of subject $i$ at time $t$ in stratum $g_i$ becomes

$$ h(t, X_i, g_i) = \sum_{g=1}^{G} I_{[g_i=g]} h_{0g}(t) \exp \left\{ \sum_{j=1}^{p} \beta_j X_{ij} \right\}, $$

where $I_A$ is an event indicator where $I_A = 1$ if $A$ occurs and $I_A = 0$ otherwise. The function $h_{0g}(t)$ represents the baseline hazard for group $g$. The coefficient $\beta_j$ represents the effect of covariate $j$. Allowing the baseline hazard to differ across strata allows flexibility often desired when proportional hazards is too strong. The SPH model can control effects of confounding variables through this stratification. The estimates of the effect of covariates remain constant across strata, so the model is still interpretable across all subjects.

2.2 Gradient Boosting for SPH

The log partial likelihood of the SPH model is

$$ \ell(\beta) = \sum_{i=1}^{n} \sum_{g=1}^{G} I_{[g_i=g]} \delta_i \left\{ X_i^\top \beta - \log \left( \sum_{\ell \in R_{ig}} \exp\{X_{i\ell}^\top \beta\} \right) \right\}, $$

where $\beta = (\beta_1, \ldots, \beta_p)^\top$, $X_i = (X_{i1}, \ldots, X_{ip})^\top$ and $R_{ig} = \{ \ell : T_\ell \geq T_i, g_i = g \}$ for all $i$ with $g_i = g$ representing the set of at risk subjects in group $g$. We adopt the following gradient boosting algorithm to find the maximum partial likelihood estimate (MPLE). Let $S_{ig}(i,j) = \sum_{\ell \in R_{ig}} X_{\ell j} \exp\{X_{\ell g}^\top \beta\}$ for $k = 0, 1, 2$. 


2.3 Stopping Criteria

Selection of the number of boosting iterations is important. Over-fitting can occur if the number of iterations is too large. The algorithm is less sensitive to the step size.

SurvBoost provides several options for optimizing the number of iterations including: $k$-fold cross validation, Bayesian information criteria, change in likelihood, or specifying the number of variables to select.

The Bayesian Information Criteria (BIC) is one approach for selecting the optimal number of boosting iterations.

$$BIC = -2 \{ l_j(\hat{\beta}_j) - l_0(\hat{\beta}_0) \} + (p_j - p_0) \log(d),$$  
(1)

where $l_j(\hat{\beta}_j)$ is the maximized likelihood for a model with $p_j$ selected variables and $l_0(\hat{\beta}_0)$ is the maximized likelihood for the reference model with $p_0$ selected variables. The number of uncensored events is $d$. [20] argue that replacing the sample size, $n$, with $d$ in the BIC calculation has better properties when dealing with censored survival models.

The extended BIC is also useful in high dimensional cases; this approach penalizes for greater complexity

$$EBIC = -2 l_j(\hat{\beta}_j) + p_j \log(d) + 2 \gamma \log \left( \frac{p_j}{p} \right),$$  
(2)

where $\left( \frac{p_j}{p} \right)$ is the size of the class of models that model $j$ belongs to, $p$ is the total number of variables. The value of $\gamma$ is fixed between 0 and 1, selected to penalize at the appropriate rate. Selecting 0 will reduce this to the standard BIC.

Cross validation is another approach which may be used to determine the stopping point. The goodness of fit function is calculated as suggested by [17]. It is the log-partial likelihood of all the data using the optimal $\beta$ determined with data excluding fold $k$ ($\beta_{-k}$) minus the log-partial likelihood excluding fold $k$ ($\ell_{-k}$) of the data with the same $\beta$.

$$CV_k(m) = -[\ell(\beta_{-k}(m)) - \ell_{-k}(\beta_{-k}(m))],$$  
(3)

Where $m$ is the current number of iterations and $k$ indicates the subset of data being excluded.

Change in likelihood is another approach incorporated in the package. This method stops iterating once a small change in likelihood, specified in the function, is reached.

$$\Delta \ell = -[\ell(\beta(m)) - \ell(\beta(m + 1))] < \alpha,$$  
(4)

Where $\alpha$ is a small constant. Default change in likelihood, used in simulations, is a change of 0.001.
3 Simulation Studies

This section compares the variable selection performance to a competing R package, *mboost*\[10\].

**Stratified Data** Stratified data was simulated such that censoring rates were relatively constant across groups and the expected survival time differed by group. These assumptions mimic realistic settings such as those encountered with data grouped by hospital or facility.

For this simulation 1,000 observations were generated into ten strata; each strata had a different baseline hazard following a Weibull distribution. The Weibull distribution shape parameter was 3 for all strata, and the scale parameter varied across strata from $e^{-1}$ to $e^{-15}$ with ten evenly spaced intervals. There were 100 true signals among 4,000 variables with true magnitude of 2 or -2. There was uniform censoring from time 0 to 200. Ten of these data sets were generated.

The following example demonstrates the importance of the stopping criteria. *SurvBoost* has five options for specifying the number of iterations as described in section 2.2. Selecting an appropriate number of iterations depends on the goals of the analysis. For example, if the goal is to achieve high sensitivity cross validation or extended BIC may be the best approach.

This simulation presents the performance of *SurvBoost* compared to the R package *mboost*. The boosting algorithm implemented in *mboost* is very similar to that of *SurvBoost* but does not allow stratification. With K-fold cross validation incorporated in *mboost*, we will compare results using cross validation and specifying a fixed number of iterations. The other stopping methods are not available in *mboost*. The performance can be compared by measures such as sensitivity and mean squared error. Table 1 presents the results of ten simulated data sets, comparing the boosting algorithm using several different stopping procedures to both default settings and cross validation methods of the package *mboost*. In this simulation, *mboost* selects fewer variables on average resulting in fewer false positives and more false negatives. Additionally the mean squared error is higher than that of all the *SurvBoost* options.

Runtime is also an important factor with this algorithm. Stratification speeds up the algorithm as seen in the first simulation. All runtimes were generated on a MacBook with 2.9GHz Intel Core i5 and 16GB memory.

| stopping method | number selected | Se  | Sp  | FDR | MSE | number of iterations | runtime (seconds) |
|------------------|-----------------|-----|-----|-----|-----|---------------------|------------------|
| SurvBoost fixed   | 110 (2)         | 0.92 (.02) | 1.00 (.00) | 0.16 (.02) | 380 (1) | 500 (0) | 44 (2) |
| mboost           | 94 (5)          | 0.78 (.03) | 1.00 (.00) | 0.17 (.03) | 387 (1) | 500 (0) | 24 (1) |
| SurvBoost cv      | 214 (13)        | 1.00 (.00) | 0.97 (.00) | 0.53 (.03) | 297 (1) | 5000 (0) | 2601 (82) |
| mboost            | 275 (8)         | 1.00 (.00) | 0.96 (.00) | 0.64 (.01) | 333 (1) | 5000 (1) | 2942 (95) |
| SurvBoost # selected | 100 (0)       | 0.85 (.03) | 1.00 (.00) | 0.15 (.03) | 384 (1) | 381 (29) | 36 (2) |
| SurvBoost likelihood | 118 (2)    | 0.96 (.01) | 0.99 (.00) | 0.18 (.02) | 375 (1) | 633 (29) | 67 (4) |
| SurvBoost EBIC    | 126 (5)         | 0.99 (.03) | 0.99 (.00) | 0.21 (.03) | 365 (1) | 998 (3) | 173 (4) |

Table 1: Results from simulation with approximately 1,500 observations in 10 strata and 4,000 variables to be selected. The table presents averages with the standard deviation, in parentheses, from ten simulated datasets. Sensitivity (Se) is calculated as the proportion of true positives out of the total number of true signals. Specificity (Sp) is calculated as the proportion of true negatives out of the total number of variables that are not true signals.

**Unstratified Data** Another simulation was used to compare performance of our method to *mboost* when stratification is not necessary for appropriate modeling. In this case one thousand observations were generated without stratification. The baseline hazard followed a Weibull distribution, with shape parameter equal to 3 and scale equal to 2. The true $\beta$ contained 100 true signals of magnitude 2 or -2 out of 1,000 variables.

We can observe in Table 2 that *SurvBoost* performs similarly to *mboost* under these conditions. *mboost* tends to select fewer variables than *SurvBoost*, so in this simulation *mboost* has fewer false positives and more false negatives compared to *SurvBoost*.
| stopping number | number of iterations | runtime (seconds) |
|-----------------|----------------------|-------------------|
| Se              | Sp                   | FDR               | MSE               | method selected |
| SurvBoost fixed | 104 (5)              | 0.78 (.02)        | 0.97 (.01)        | 0.25 (.04)      | 379 (1)         | 500 (0) | 4 (1) |
| mboost          | 82 (5)               | 0.64 (.02)        | 0.98 (.00)        | 0.22 (.04)      | 387 (0)         | 500 (0) | 10 (0) |
| SurvBoost cv    | 213 (13)             | 1.00 (.00)        | 0.87 (.01)        | 0.53 (.03)      | 299 (2)         | 5000 (0) | 391 (24) |
| mboost          | 181 (13)             | 1.00 (.01)        | 0.91 (.01)        | 0.45 (.04)      | 333 (2)         | 5000 (1) | 1222 (44) |
| SurvBoost # selected | 100 (0)            | 0.76 (.03)        | 0.97 (.00)        | 0.24 (.03)      | 380 (2)         | 453 (42) | 4 (0) |
| SurvBoost likelihood | 108 (5)            | 0.81 (.03)        | 0.97 (.01)        | 0.25 (.03)      | 377 (1)         | 549 (18) | 6 (0) |
| SurvBoost EBIC  | 38 (1)               | 0.29 (.01)        | 0.99 (.00)        | 0.09 (.25)      | 389 (0)         | 300 (2) | 13 (0) |

Table 2: Results from simulation with approximately 1,000 observations and 1,000 variables to be selected. The table presents averages with the standard deviation from ten simulated datasets.

4 Illustration of Package

This section provides a brief tutorial on how to use this package based on simulated data. In order to install the package, several other R packages must be installed. The code relies on Rcpp, RcppArmadillo, and RcppParallel in order to improve computational speed. Additionally the survival package is used for simulation and post selection inference and will be required for installation of SurvBoost. The following line of R code installs the package.

```
install.packages("SurvBoost_0.1.0.tar.gz", type="source", repos = NULL)
```

4.1 Model fitting

The boosting_core() function requires similar inputs to the familiar coxph() function from the package survival. boosting_core(formula, data = matrix(), rate = 0.01, control = 500, ...) The input formula has the form Surv(time, death) ~ variable1 + variable2. The input data is in matrix form or a data frame. Two additional parameters must be specified for the boosting algorithm: rate and control. Rate is the step size in the algorithm, although choice of this may not impact the performance too significantly [4], default value is set to 0.01. Selecting an appropriate number of iterations to run the algorithm will, however, have a greater impact on the results. The last input control is used to determine the number of iterations to run the algorithm, default value is 500.

| Call Method | Description |
|-------------|-------------|
| boosting_core(formula, data) | fixed mstop = 500 |
| boosting_core(formula, data, control=1000) | fixed mstop = specified value |
| boosting_core(formula, data, control_method="cv") | 10-fold cross validation |
| boosting_core(formula, data, control_method="num_selected", control_parameter = 5) | number selected, need to specify number of variables |
| boosting_core(formula, data, control_method="likelihood") | change in likelihood |
| boosting_core(formula, data, control_method="BIC") | minimum BIC or EBIC |
| boosting_core(formula, data, control_method="AIC") | minimum AIC |

Table 3: Stopping criteria options for boosting_core function.

| Function | Result |
|----------|--------|
| summary.boosting() | prints summary of variable selection and estimation |
| modelfit.boosting() | prints summary of model and data |
| plot.boosting() | plots variable selection frequency |
| predict.boosting() | generates predicted hazard ratio for each observation or a new dataset |

Table 4: Functions available in SurvBoost package. Every function accepts a boosting object input to generate the corresponding result.
4.2 Simple example

We present a simple example demonstrating the convenience of using the package for stratified data. We simulate survival data for five facilities with different constant baseline hazards.

```r
R > TrueBeta
[1] 0.5 0.5 0.0 0.0 0.0 -0.5 0.5 0.0 0.0
R > set.seed(123)
R > data_small <- simulate_survival_cox(true_beta=TrueBeta,
          base_hazard="auto",
          num_facility=5,
          input_facility.size=100, cov.structure="ar",
          block_size=5, rho=0.6, censor_dist="unif",
          censor_const=2, tau=Inf, normalized=F)
```

We have \( p = 10 \) and \( |\beta_j| \) ranges from 0 to 0.5. There are five “facilities” with average size of 100, and \( n \) is approximately 500. The covariance structure within the blocks is AR(1) with correlation 0.6. The censoring rate is about 33\%. In this case the variable `facility_idx` indicates the variable to stratify on in the survival model; each “facility” in this simulated data has a different baseline hazard function.

Another feature of the package assists with determining variables to stratify on if this information is unknown. The function `strata.boosting` will print box plots and a summary table of the survival time grouped by splits in a the specified variable. The variable can be categorical or continuous; if continuous, the function will split on the median value to demonstrate whether there appears to be a difference in the survival time distribution for the two groups.

```r
R > strata.boosting(data_small$facility_idx, data_small$time)
```

![Box plots of survival time by facility index in simulated data generated by the function `strata.boosting`.](image)

**Figure 1:** Box plots of survival time by facility index in simulated data generated by the function `strata.boosting`. 
Simulated data includes a vector of survival or censoring time, \( time \), indicator of an event, \( delta \), and matrix of covariates, \( Z \). Then generate the formula including all possible variables for selection.

\[
\begin{align*}
R & \geq \text{simulated data} \\
R & \Rightarrow \text{time} \leftarrow \text{data_small}$time \\
R & \Rightarrow \text{delta} = \text{data_small}$delta \\
R & \Rightarrow Z \leftarrow \text{as.matrix(data_small[, -c(1, 2, 3)]}) \\
R & \Rightarrow \text{covariates} \leftarrow \text{paste("strata(facility_idx)", paste(colnames(Z), collapse = "+")}) \\
R & \Rightarrow \text{formula} \leftarrow \text{as.formula(paste("Surv(time, delta) ~", covariates))}
\end{align*}
\]

Run the \texttt{boosting.core()} function to obtain the variables selected. This example uses the number of iterations control as a fixed input of 75 and update rate of 0.1.

\[
\begin{align*}
R & \geq \text{test1} \leftarrow \text{boosting.core(formula,} \\
R & \text{data= data_small,} \\
R & \text{rate } = 0.1,} \\
R & \text{control } = 75) \\
R & \Rightarrow \text{summary.boosting(test1)} \\
\end{align*}
\]

Function \texttt{summary.boosting()} displays the variables which are selected as well as the coefficient estimates and the number of boosting iterations performed. Set the argument \texttt{all.beta = TRUE} to see all the variables, not just those selected. More detailed information about the model can be obtained through the function \texttt{modelfit.boosting()}.

\[
\begin{align*}
R & \geq \text{test2} \leftarrow \text{boosting.core(formula,} \\
R & \text{data= data_small,} \\
R & \text{rate } = 0.1,} \\
R & \text{control_method } = \text{num_selected,} \\
R & \text{control.parameter } = 5) \\
R & \Rightarrow \text{summary.boosting(test2)} \\
\end{align*}
\]

To use a different method for the number of boosting iterations use the arguments \texttt{control_method} and \texttt{control_parameter}. For example,

\[
\begin{align*}
R & \geq \text{test2} \leftarrow \text{boosting.core(formula,} \\
R & \text{data= data_small,} \\
R & \text{rate } = 0.1,} \\
R & \text{control_method } = \text{num_selected,} \\
R & \text{control.parameter } = 5) \\
R & \Rightarrow \text{summary.boosting(test2)} \\
\end{align*}
\]

This option iterates until the specified number of variables, 5 in this example, are selected. See methods for other stopping criteria.
The `plot.boosting()` function displays a plot of the selection frequency by the number of iterations. Another option of the `plot.boosting()` function is to plot the coefficient paths of each variable by the number of boosting iterations. See Figures 2 and 3.

![Figure 2: Plot generated by plot.boosting function, variable selection frequency by number of boosting iterations.](image)

Figure 2: Plot generated by plot.boosting function, variable selection frequency by number of boosting iterations.

![Figure 3: Plot generated by plot.boosting function with option “coefficients”, coefficient paths for variables selected by number of boosting iterations.](image)

Figure 3: Plot generated by plot.boosting function with option “coefficients”, coefficient paths for variables selected by number of boosting iterations.

The function `predict.boosting()` provides an estimate of the hazard ratio for each observation in the dataset provided relative to the average of \( p \) predictors.

```r
R > predict.boosting(test1)[1:6]  
 46.385476  1.823920  42.049932  16.427860  4.013200  2.243711
```

The model selected using boosting can be refit with `coxph()` for post selection inference. The function `inference.boosting()` will perform this refitting and output the coefficient estimates with corresponding standard errors and p-values.

```r
R > fmla <- summary.boosting(test1)$formula
```
5 TCGA Data Example

Data from three breast cancer cohorts was used to demonstrate this method on data outside of the simulation framework. There were 578 patients included in the combined data, with 8864 variables measured for each patient: 8859 genes and 5 phenotypic variables. The phenotype variables included age at diagnosis, tumor size, cancer stage, progesterone-receptor status, and estrogen-receptor status. The data can be downloaded from The Cancer Genome Atlas (TCGA) [16, 7, 15].

The patients were split into two cohorts depending on their cancer stage and tumor size. One cohort contained patients with the less severe prognosis, cancer stage of one and tumor size less than the median; the other cohort contained those with cancer stage greater than one and/or with a tumor larger than the median size.

R> fit.plot <- survfit(Surv(survival_time, survival_ind) ~ as.factor(severity), data=data)
R> ggsurvplot(fit.plot, conf.int = TRUE, risk.table = TRUE, risk.table.col="strata", ggtheme = theme_bw(), palette = "grey")
Figure 4: Survival curves for the two strata based on cancer stage and tumor size. This plot demonstrates that the proportional hazards assumption may not hold in this case. Stratifying based on this criteria generates the following results.

Using stability selection \[14\], 14 variables were identified with selection frequencies greater than 50% from 50 iterations of subsampling. Age and progesterone-receptor status were selected in addition to 12 genes. The boosting algorithm was performed with the number of iterations fixed at the sample size of 578.

Figure 5: Selection frequencies for genes or phenotype variables that were selected at least 50% of the time with stability selection.

Several of the genes selected in this analysis have been previously identified as having an association with breast cancer. Psoriasin (S100A7) has been associated with breast cancer \[1\]. Several studies have found COL2A1 to be part of gene signatures for predicting tumor recurrence \[22, 21\]. Other genes selected that have been identified as part of a gene signature or association with breast cancer tumor progression risk include: ZIC1 \[3\], CYP2B6 \[19\], ELF5 \[18\], IGJ \[3\], DHR52 \[13\], and CEACAM5 \[2\]. Mboost using the same criteria but without a stratified model only identifies one gene of importance, MC2R, demonstrating the utility of the SPH model in this context.
6 Conclusion

In this article, we introduce a new R package SurvBoost which implements the gradient boosting algorithm for high-dimensional variable selection in the stratified proportional hazards (SPH) model, while most existing R packages, such as mboost only focus on the proportional hazards model. In the simulation studies, we show that SurvBoost can improve the model fitting and achieve better variable selection accuracy for the data with stratified structures. In addition, we optimize the implementations of the gradient boosting in both the SPH and the PH models. For the PH model fitting, SurvBoost can reduce about 30%-50% computational time compared to mboost. In the future, we plan to extend the package to handle more complex survival data such as left-truncation data and interval censoring data.

References

[1] Sahar Al-Haddad, Zi Zhang, Etienne Leygue, Linda Snell, Aihua Huang, Yulian Niu, Tamara Hiller-Hitchcock, Kate Hole, Leigh C. Murphy, and Peter H. Watson. Psoriasin (s100a7) expression and invasive breast cancer. *The American Journal of Pathology*, 155(6):2057–2066, 1999.

[2] Rosalyn D. Blumenthal, Evelyn Leon, Hans J. Hansen, and David M. Goldenberg. Expression patterns of ceacam5 and ceacam6 in primary and metastatic cancers. *BMC Cancer*, 7(1):2, Jan 2007.

[3] Brenda J. Boersma, Mark Reimers, Ming Yi, Joseph A. Ludwig, Brian T. Lake, Robert M. Stephens, Harry G. Yfantis, Dong H. Lee, John N. Weinstein, and Stefan Ambs. A stromal gene signature associated with inflammatory breast cancer. *International Journal of Cancer*, 122(6):1324–1332, 2008.

[4] Peter Bühlmann and Torsten Hothorn. Boosting algorithms: Regularization, prediction and model fitting (with discussion). *Statistical Science*, 22(4):477–505, 2007.

[5] Peter Bühlmann and Bin Yu. Boosting. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):69–74, 2010.

[6] Rumela Chakrabarti, Julie Hwang, Mario Andres Blanco, Yong Wei, Martin Lukacisin, Rose-Anne Romano, Kirsten Smalley, Song Liu, Qifeng Yang, Toni Ibrahim, Laura Mercatali, Dino Amadori, Bruce G. Haffty, Satrajit Sinha, and Yibin Kang. Elf5 inhibits the epithelial-mesenchymal transition in mammary gland development and breast cancer metastasis by transcriptionally repressing snail2. *Nature Cell Biology*, 14:1212–1222, 2012.

[7] Koei Chin, Sandy DeVries, Jane Fridlyand, Paul T. Spellman, Ritu Roydasgupta, Wen-Lin Kuo, Anna Lapuk, Richard M. Neve, Zuwei Qian, Tom Ryder, Faqiong Chen, Heidi Feiler, Taku Tokuyasu, Chris Kingsley, Shanaz Dairkee, Zhenhang Meng, Karen Chew, Daniel Pinkel, Ajay Jain, Britt Marie Ljung, Laura Esserman, Donna G. Albertson, Frederic M. Waldman, and Joe W. Gray. Genomic and transcriptional aberrations linked to breast cancer pathophysiology. *Cancer Cell*, 10(6):529–541, 2006.

[8] Jelle J. Goeman. L1 penalized estimation in the cox proportional hazards model. *Biometrical Journal*, 52(1):70–84, 2010.

[9] Kevin He, Yanming Li, Ji Zhu, Hongliang Liu, Jeffrey E. Lee, Christopher I. Amos, Terry Hyslop, Jiashun Jin, Huazhen Lin, Qinyi Wei, and Yi Li. Component-wise gradient boosting and false discovery control in survival analysis with high-dimensional covariates. *Bioinformatics*, 32(1):50–57, 2016.

[10] Benjamin Hofner, Andreas Mayr, Nikolay Robinzonov, and Matthias Schmid. Model-based boosting in R: A hands-on tutorial using the R package mboost. *Computational Statistics*, 29:3–35, 2014.

[11] Torsten Hothorn, Peter Bühlmann, Thomas Kneib, Matthias Schmid, and Benjamin Hofner. Model-based boosting 2.0. *Journal of Machine Learning Research*, 11:2109–2113, 2010.

[12] Wenzin Jiang. Process consistency for adaboost. *Ann. Statist.*, 32(1):13–29, 02 2004.

[13] Oscar Krijgsman, Paul Roepman, Wilbert Zwart, Jason S. Carroll, Sun Tian, Femke A. de Snoo, Richard A. Bender, Paul Bernards, and Annuska M. Glas. A diagnostic gene profile for molecular subtyping of breast cancer associated with treatment response. *Breast Cancer Research and Treatment*, 133(1):37–47, May 2012.

[14] Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.
[15] Lance D. Miller, Johanna Smeds, Joshy George, Vinsensius B. Vega, Liza Vergara, Alexander Ploner, Yudi Pawitan, Per Hall, Sigrid Klaar, Edison T. Liu, and Jonas Bergh. An expression signature for p53 status in human breast cancer predicts mutation status, transcriptional effects, and patient survival. *Proceedings of the National Academy of Sciences of the United States of America*, 102(38):13550–13555, 2005.

[16] A Naderi, A E Teschendorff, N L Barbosa-Morais, S E Pinder, A R Green, D G Powe, J F R Robertson, S Aparicio, I O Ellis, J D Brenton, and C Caldas. A gene-expression signature to predict survival in breast cancer across independent data sets. *Oncogene*, 26:1507–1516, 08 2006.

[17] Noah Simon, Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for cox’s proportional hazards model via coordinate descent. *Journal of Statistical Software, Articles*, 39(5):1–13, 2011.

[18] Robert Tibshirani. The lasso method for variable selection in the cox model. *Statistics in Medicine*, 16(4):385–395, 1997.

[19] S Tozlu, I Girault, S Vacher, J Vendrell, C Andrieu, F Spyratos, P Cohen, R Lidereau, and I Bieche. Identification of novel genes that co-cluster with estrogen receptor alpha in breast tumor biopsy specimens, using a large-scale real-time reverse transcription-pcr approach. *Endocrine-Related Cancer*, 13(4):1109–1120, 2006.

[20] Chris T. Volinsky and Adrian E. Raftery. Bayesian information criterion for censored survival models. *Biometrics*, 56(1):256–262, 2000.

[21] Yixin Wang, Jan GM Klijn, Yi Zhang, Anieta M Sieuwerts, Maxime P Look, Fei Yang, Dmitri Talantov, Mieke Timmermans, Marion E Meijer van Gelder, Jack Yu, Tim Jatkoe, Els MJJ Berns, David Atkins, and John A Foekens. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *The Lancet*, 365(9460):671–679, 2005.

[22] Jack X. Yu, Anieta M. Sieuwerts, Yi Zhang, John WM Martens, Marcel Smid, Jan GM Klijn, Yixin Wang, and John A. Foekens. Pathway analysis of gene signatures predicting metastasis of node-negative primary breast cancer. *BMC Cancer*, 7(1):182, 2007.
Emily Morris
Department of Biostatistics
University of Michigan
1415 Washington Heights, Ann Arbor MI 48109
E-mail: emorrisl@umich.edu

Jian Kang
Department of Biostatistics
University of Michigan
1415 Washington Heights, Ann Arbor MI 48109
E-mail: jiankang@umich.edu