Survival Prediction Using Gene Expression Data

Zhu Wang
UT Health San Antonio

Abstract

This document describes applications of R package bujar for predicting survival in diffuse large B cell lymphoma treated with chemotherapy plus Rituximab using gene expression data.

Keywords: survival, Buckley-James regression, prediction, boosting, variable selection.

1. Introduction

Researchers have been interested in predicting survival in diffuse large B cell lymphoma (DLBCL) treated with chemotherapy. We re-evaluate clinical and microarray data in Lenz et al. (2008). Data from two treatment plans were collected: CHOP and R-CHOP. CHOP is a combination chemotherapy with cyclophosphamide, doxorubicin, vincristine and prednisone. The current gold standard includes rituxima immunotherapy in addition to the chemotherapy (R-CHOP). It was interesting to identify genes that predict survival among patients who received CHOP also retain their prognostic power among patients who received R-CHOP. Data from 181 CHOP patients (training data) are used to build predictive models and data from 233 R-CHOP patients (test data) are used to validate the models. Due to the nature of high-dimensional data with 54675 probe sets or covariates, we first conduct a pre-selection procedure on the training data. This procedure filters out genes with lower variations if a sample variance for a gene was smaller than the 10th percentile for that gene. Details on how the data were read and pre-screened may be found in dlbcl_raw.R in the vignettes directory.

Afterwards we have 3833 remaining probe sets which were stored in the R package bujar (Wang 2015). Test data with the same remaining genes is used for validation.

One of the challenges is that we have right censored data, as often occurred in survival data analysis. Although proportional-hazards regression is a popular option, we focus on Buckley-James (BJ) regression for high-dimensional data proposed by Wang and Wang (2010). Essentially, BJ regression iteratively imputes the censored survival outcomes using the conditional expectation obtained from the current estimates, and the imputed survival outcomes are re-fitted with regression techniques, which in turn lead to different BJ predictive models depend on what regression model is chosen. We evaluate BJ boosting algorithms and BJ penalized regression implemented in bujar. To avoid overfitting, predictive modeling typically requires tuning parameter(s) selection. Tuning parameter is selected by data-driven cross-validation, unless otherwise specified. However, if twin boosting is conducted, the boosting iteration number is fixed in the second round of boosting. The analysis results can be different from Wang and Wang (2010) for two reasons. First, computer codes have been changed for BJ boost-
Survival Prediction

ing with componentwise least squares or smoothing splines as base learner. In the current implementation since version 0.1-10, every BJ iteration uses the optimal boosting iteration obtained from the last BJ iteration if tuning parameter is selected. The change results in more sparse variable selection and also consistent with BJ boosting with trees as base learner. Second, BJ boosting (not twin boosting) with trees involves an internal random mechanism for tree model building. In the current implementation since version 0.1-10, a new parameter \texttt{rng} in function \texttt{bujar} can be used for reproducible results.

\begin{verbatim}
R> library("bujar")
R> data("chop")
R> data("rchop")
R> ###censoring rate in the CHOP data
R> sum(chop$status==0)/nrow(chop)

[1] 0.4199

R> rchop <- subset(rchop, select=colnames(rchop)%in% colnames(chop))
R> chop$survtime <- chop$survtime + 1 ### add 1 for log-transformation
\end{verbatim}

2. BJ Boosting with Linear Regression

2.1. BJ boosting with componentwise least squares (BJ-LS)

\begin{verbatim}
R> set.seed(123)
R> res.lin <- bujar(y=log(chop[,1]), cens=chop[,2], x=chop[,-(1:2)], tuning=TRUE,
                      cv=TRUE, n.cores=1, mstop=1000)
R> ###number of genes selected with BJ-LS
R> sum(res.lin$xselect==1)

[1] 27

R> coef.bj <- coef(res.lin)
R> ###estimated non-zero coefficients (only list 10)
R> coef.bj[abs(coef.bj)>0][1:10]

   (Intercept)  x1553499_s_at  x1555939_at  x1558813_at  x1558999_x_at  x1561016_at
   -0.41908     0.01159     0.01139     0.01112    -0.04109    -0.02474
    x1562727_at  x1565740_at  x1565861_at  x1568732_at
   -0.01287     0.02801     0.01078     0.03484
\end{verbatim}

R> library("survival")
R> cutyear <- 3
R> pred.bj <- predict(res.lin, newx=rchop[, -(1:2)])
R> pred.bj <- exp(pred.bj) - 1
R> group <- cut(pred.bj, breaks=c(-1, cutyear, 100), labels=c("high","low"))
R> dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
R> fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
R> fit.diff

Call:
survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)

    N Observed Expected (O-E)^2/E (O-E)^2/V
group=high 33    16   6.25   15.21   17.2
group=low 200   44  53.75    1.77   17.2

Chisq= 17.2 on 1 degrees of freedom, p=3e-05

R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km )
R> plot(fit.surv, xlab="Year past therapy",ylab="Survival probability", lty = 1:2, col=c("red","blue"), mark.time=TRUE)
R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))

![Kaplan-Meier survival curves for BJ-LS regression.](image)

Figure 1: Kaplan-Meier survival curves for BJ-LS regression.
2.2. BJ twin boosting with componentwise least squares

```r
R> res.lin2 <- bujar(y=log(chop[,1]), cens=chop[,2], x=chop[-(1:2)], tuning=TRUE,
>             cv=FALSE, mstop=1000, twin=TRUE, mstop2=100)
R> ### number of genes selected with BJ-LS
R> sum(res.lin2$xselect==1)

[1] 10

R> coef.bj <- coef(res.lin2)
R> coef.bj[abs(coef.bj)>0]

(Intercept) 1558999_x_at 1561016_at 1562727_at 1568732_at 1569344_a_at
    -0.649229  -0.129651  -0.085982  -0.046537   0.153564   0.008244
  212713_at 229839_at 237797_at 242758_x_at 244346_at
     0.044846   0.128520   0.210326   0.141115   0.124829

R> pred.bj <- predict(res.lin2, newx=rchop[-(1:2)])
R> pred.bj <- exp(pred.bj) - 1
R> group <- cut(pred.bj, breaks=c(-1, cutyear, 100), labels=c("high", "low"))
R> dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
R> fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
R> fit.diff

Call:
survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)

    N Observed Expected (O-E)^2/E (O-E)^2/V
group=high  65     24     15   5.45  7.32
group=low 168     36     45   1.81  7.32

   Chisq= 7.3 on 1 degrees of freedom, p= 0.007
```
R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km )
R> plot(fit.surv, xlab="Year past therapy",ylab="Survival probability",
    lty = 1:2, col=c("red","blue"), mark.time=TRUE)
R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))

Figure 2: Kaplan-Meier survival curves for twin BJ-LS regression.
To reduce computing burden for other modeling methods, we adopted a supervised gene screening to select the top 100 probe sets based on univariate BJ regression.

```r
library("rms")
res <- rep(NA, ncol(chop))
for(i in 3:ncol(chop)){
  bjres <- try(bj(Surv(survtime, status) ~ chop[,i], data=chop, link="log"))
  ### if BJ convergence fails, still included for further analysis
  if(inherits(bjres, "try-error")) res[i] <- 1e-5
  else res[i] <- anova(bjres)[1,3] # p-value
}
nsel <- 100
### select top nsel=100 genes with most significant p-values
chop2 <- chop[, c(1, 2, sort.list(res, decreasing=FALSE)[1:nsel])]
rchop2 <- rchop[, c(1, 2, sort.list(res, decreasing=FALSE)[1:nsel])]
colnames(chop2)[-c(1:2)] <- colnames(rchop2)[-c(1:2)] <-
paste("x", colnames(chop2)[-c(1:2)], sep="")
detach(package:rms)
```

### 3. BJ LASSO

Within each BJ iteration, the LASSO is used to fit the imputed survival outcomes. The penalty tuning parameter is fixed at the 20th value in 100 penalty sequence values determined within each BJ iteration.

```r
res.lasso <- bujar(y=log(chop2[,1]), cens=chop2[,2], x=chop2[-c(1:2)],
  learner="enet2", tuning=FALSE, whichlambda=20)
### how many genes selected by BJ-LASSO
sum(res.lasso$xselect==1)
[1] 44
```

```r
### estimated non-zero coefficients (only list 10)
coef.bj <- coef(res.lasso)
coef.bj[abs(coef.bj)>0][1:10]
```

```r
(Intercept) x229839_at x243713_at x1553317_s_at x244434_at x240898_at
-0.672767 0.050211 0.033440 0.018877 0.004576 0.010742
x206439_at x1569344_a_at x237493_at x237797_at
0.017642 0.020257 0.001879 0.104734
```

```r
pred.bj <- predict(res.lasso, newx=rchop2[-c(1:2)])
pred.bj <- exp(pred.bj) - 1
group <- cut(pred.bj, breaks=c(-1, cutyear, 100), labels=c("high", "low"))
dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
fit.diff
```
Call:
\[
\text{survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)}
\]

\[
\begin{array}{cccccc}
\text{group} & \text{N} & \text{Observed} & \text{Expected} & (O-E)^2/E & (O-E)^2/V \\
\text{high} & 36 & 18 & 7.08 & 16.86 & 19.3 \\
\text{low} & 197 & 42 & 52.92 & 2.25 & 19.3 \\
\end{array}
\]

Chisq = 19.3 on 1 degrees of freedom, p = 1e-05

\[
\text{R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km)}
\]

\[
\text{R> plot(fit.surv, xlab="Year past therapy", ylab="Survival probability", lty = 1:2, col=c("red","blue"), mark.time=TRUE)}
\]

\[
\text{R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))}
\]

Figure 3: Kaplan-Meier survival curves for BJ-LASSO regression
4. BJ SCAD

BJ-SCAD is an extension of BJ-LASSO (Wang and Wang 2010; Fan and Li 2001). Within each BJ iteration, the SCAD is used with the imputed survival outcomes. The penalty tuning parameter is fixed at the 20th value in 100 penalty sequence values determined within each BJ iteration.

R> res.scad <- bujar(y=log(chop2[,1]), cens=chop2[,2], x=chop2[,-(1:2)], learner="snet", tuning=FALSE, whichlambda=20)
R> ### how many genes selected by BJ-SCAD
R> sum(res.scad$xselect==1)

[1] 34

R> ### estimated non-zero coefficients (only list 10)
R> coef.bj <- coef(res.scad)
R> coef.bj[abs(coef.bj)>0][1:10]

(Intercept) x229839_at x243713_at x1553317_s_at x244434_at x240898_at
-0.963676 0.096681 0.026108 0.010459 0.003088 0.005155
x1569344_a_at x237797_at x212713_at x1569100_a_at
0.021854 0.188881 0.045009 0.018037

R> pred.bj <- predict(res.scad, newx=rchop2[,-(1:2)])
R> pred.bj <- exp(pred.bj) - 1
R> group <- cut(pred.bj, breaks=c(-1, cutyear, 100), labels=c("high", "low"))
R> dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
R> fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
R> fit.diff

Call:
survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)

     N Observed Expected (O-E)^2/E (O-E)^2/V
group=high 45    23    8.87    22.5    26.8
group=low 188  37   51.13     3.9    26.8

  Chisq= 26.8 on 1 degrees of freedom, p= 2e-07
R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km)
R> plot(fit.surv, xlab="Year past therapy", ylab="Survival probability",
       lty = 1:2, col=c("red","blue"), mark.time=TRUE)
R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))

Figure 4: Kaplan-Meier survival curves for BJ-SCAD regression.
5. BJ Boosting with Smoothing Splines

5.1. BJ boosting with componentwise smoothing splines (BJ-SM)

```r
R> set.seed(123)
R> res.ss <- bujar(y=log(chop2[,1]), cens=chop2[,2], x=chop2[,-(1:2)],
                  learner="pspline", tuning=FALSE, cv=FALSE, mstop=100)
R> ## how many genes selected by BJ smoothing splines, only list 10
R> sum(res.ss$xselect==1)
[1] 36
R> colnames(res.ss$x)[res.ss$xselect==1][1:10]
[1] "x229839_at" "x243713_at" "x244434_at" "x240898_at" "x226869_at"
[6] "x237493_at" "x204879_at" "x231442_at" "x208168_s_at" "x231049_at"
R> pred.bj <- predict(res.ss, newx=rchop2[,-(1:2)])
R> pred.bj <- exp(pred.bj) - 1
R> group <- cut(pred.bj, breaks=c(-1, cutyear, 100), labels=c("high", "low"))
R> dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
R> fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
R> fit.diff

Call:
survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)

        N Observed Expected (O-E)^2/E (O-E)^2/V
group=high  48    20    9.79    10.64    12.8
group=low 185    40   50.21    2.08    12.8

Chisq= 12.8 on 1 degrees of freedom, p= 3e-04
```
R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km)
R> plot(fit.surv, xlab="Year past therapy",ylab="Survival probability", lty = 1:2, col=c("red","blue"), mark.time=TRUE)
R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))

Figure 5: Kaplan-Meier survival curves for BJ-SM regression.
5.2. BJ twin boosting with componentwise smoothing splines

```R
R> set.seed(123)
R> res.ss2 <- bujar(y=log(chop2[,1]), cens=chop2[,2], x=chop2[,-(1:2)],
   learner="pspline", tuning=TRUE, cv=TRUE, n.cores=1, mstop=100, twin=TRUE, mstop2=200)
R> ### how many genes selected by BJ twin smoothing splines, only list 10
R> sum(res.ss2$xselect==1)
[1] 25
R> colnames(res.ss2$x)[res.ss2$xselect==1][1:10]
[1] "x229839_at" "x243713_at" "x1553317_s_at" "x244434_at" "x240898_at"
[6] "x206439_at" "x1569344_a_at" "x237797_at" "x212713_at" "x1569100_a_at"
R> pred bj <- predict(res.ss2, newx=rchop2[,-(1:2)])
R> pred bj <- exp(pred bj) - 1
R> group <- cut(pred bj, breaks=c(-1, cutyear, 100), labels=c("high", "low"))
R> dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
R> fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
R> fit.diff

Call:
survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)

    N Observed Expected (O-E)^2/E (O-E)^2/V
group=high 62     26     13.2    12.4  16.1
group=low 171    34     46.8    3.5  16.1

  Chisq= 16.1 on 1 degrees of freedom, p= 6e-05
```
R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km)
R> plot(fit.surv, xlab="Year past therapy",ylab="Survival probability", lty = 1:2, col=c("red","blue"), mark.time=TRUE)
R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))

Figure 6: Kaplan-Meier survival curves for twin BJ-SM regression.
6. BJ Boosting with Regression Trees

6.1. BJ boosting with regression stumps (BJ-Tree)

```r
R> res.tree1 <- bujar(y=log(chop2[,1]), cens=chop2[,2], x=chop2[,-(1:2)],
                     learner="tree", tuning=TRUE, cv=TRUE, mstop=1000, n.cores=1, rng=123)
R> ###Number of genes selected with tree, only list 10
R> sum(res.tree1$xselect==1)
[1] 75
R> colnames(res.tree1$x)[res.tree1$xselect==1][1:10]
[1] "x229839_at" "x243713_at" "x1553317_s_at" "x1554413_s_at" "x244434_at"
[6] "x240898_at" "x206439_at" "x226869_at" "x1569344_a_at" "x237493_at"
R> pred.bj <- predict(res.tree1, newx=rchop2[,-(1:2)])
R> pred.bj <- exp(pred.bj) - 1
R> group <- cut(pred.bj, breaks=c(-1, cutyear, 100), labels=c("high", "low"))
R> dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
R> fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
R> fit.diff
Call:
survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)

    N Observed Expected (O-E)^2/E (O-E)^2/V
group=high  32   13  6.29  7.173  8.1
 group=low 201   47 53.71  0.839  8.1

     Chisq= 8.1 on 1 degrees of freedom, p= 0.004
```
R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km )
R> plot(fit.surv, xlab="Year past therapy",ylab="Survival probability",
   lty = 1:2, col=c("red","blue"), mark.time=TRUE)
R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))

Figure 7: Kaplan-Meier survival curves for BJ-Tree regression.
6.2. BJ twin boosting with regression stumps

```r
R> res.tree2 <- bujar(y=log(chop2[,1]), cens=chop2[,2], x=chop2[,-(1:2)],
  learner="tree", tuning=TRUE, cv=TRUE, mstop=1000, twin=TRUE, mstop2=100,
  n.cores=1, rng=123)
R> ###Number of genes selected with tree, only list 10
R> sum(res.tree2$xselect==1)
[1] 29
R> colnames(res.tree2$x)[res.tree2$xselect==1][1:10]
[1] "x229839_at" "x244434_at" "x240898_at" "x226869_at" "x237493_at"
[6] "x237797_at" "x204879_at" "x231442_at" "x1556395_at" "x1553499_s_at"
```

```r
R> pred.bj <- predict(res.tree2, newx=rchop2[,-(1:2)])
R> pred.bj <- exp(pred.bj) - 1
R> group <- cut(pred.bj, breaks=c(-1, cutyear, 100), labels=c("high", "low"))
R> dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
R> fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
R> fit.diff
```

```
Call:
  survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)

                           N Observed Expected (O-E)^2/E (O-E)^2/V
  group=high  74     31   15.4     15.66       21.4
  group=low 159     29   44.6     5.43       21.4

    Chisq= 21.4 on 1 degrees of freedom, p= 4e-06
```
R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km )
R> plot(fit.surv, xlab="Year past therapy",ylab="Survival probability", lty = 1:2, col=c("red","blue"), mark.time=TRUE)
R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))

Figure 8: Kaplan-Meier survival curves for twin BJ-Tree regression.
6.3. BJ boosting with regression trees of degree 4

R> res.tree4 <- bujar(y=log(chop2[,1]), cens=chop2[,2], x=chop2[,-(1:2)],
  learner="tree",degree=4, tuning=TRUE, cv=TRUE, mstop=100, rel.inf=TRUE,
  n.cores=1,rng=123)
R> ###Number of genes selected with tree, only list 10
R> sum(res.tree4$xselect==1)

[1] 77

R> colnames(res.tree4$x)[res.tree4$xselect==1][1:10]

[1] "x229839_at"  "x243713_at"  "x1553317_s_at"  "x244434_at"  "x240898_at"
[6] "x206439_at"  "x226869_at"  "x1569344_a_at"  "x237797_at"  "x204879_at"

R> pred.bj <- predict(res.tree4, newx=rchop2[,-(1:2)])
R> pred.bj <- exp(pred.bj) - 1
R> group <- cut(pred.bj, breaks=c(-1, cutyear, 100), labels=c("high", "low"))
R> dat.km <- data.frame(survtime=rchop$survtime, status = rchop$status, group=group)
R> fit.diff <- survdiff(Surv(survtime, status) ~ group, data=dat.km)
R> fit.diff

Call: survdiff(formula = Surv(survtime, status) ~ group, data = dat.km)

    N Observed Expected (O-E)^2/E (O-E)^2/V
  group=high 35     15  6.9435   9.3696   10.7
  group=low 198     45 53.0628   1.2194   10.7

  Chisq= 10.7  on 1 degrees of freedom, p= 0.001
R> fit.surv <- survfit(Surv(survtime, status) ~ group, data=dat.km)
R> plot(fit.surv, xlab="Year past therapy", ylab="Survival probability",
       lty = 1:2, col=c("red","blue"), mark.time=TRUE)
R> legend(1, .1, c("High risk", "Low risk"), lty = 1:2, col=c("red","blue"))

Figure 9: Kaplan-Meier survival curves for BJ-Tree (degree 4) regression.
Partial dependence plots can be utilized to show the impact of one or more covariates on the response after taking account the average effects of all other covariates in the model. We can generate plots for selected genes. See Figure 5 in Wang and Wang (2010).

```r
R> gene <- c("x1558999_x_at", "x212713_at", "x224043_s_at", "x229839_at", 
             "x237515_at", "x237797_at", "x242758_x_at", "x244346_at")
R> library("gridExtra")
R> for(i in 1:length(gene))
  eval(parse(text=(paste("a", i, " <- plot(res.tree4$res.fit, 
                   i.var=which(colnames(res.tree4$x) == gene[i])", sep=""))
R> grid.arrange(a1, a2, a3, a4, a5, a6, a7, a8, ncol=4)
```

![Partial plots of selected genes based on BJ-Tree (degree=4) regression.](image)

Figure 10: Partial plots of selected genes based on BJ-Tree (degree=4) regression.
The two-way interaction partial plots display gene-gene interactions similar to Figure 6 in Wang and Wang (2010).

R> for(i in 1:6)
  eval(parse(text=(paste("b", i, " <- plot(res.tree4$res.fit,
    i.var=unlist(res.tree4$interactions$rank.list[i,c(1, 3)])",
    sep="")))))
R> grid.arrange(b1, b2, b3, b4, b5, b6, ncol=2)

Figure 11: Gene-gene interactions based on BJ-Tree (degree=4) regression.
7. SessionInfo

R> sessionInfo();

R version 3.5.2 (2018-12-20)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 18.04.1 LTS

Matrix products: default
BLAS: /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.7.1
LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.7.1

locale:
[1] LC_COLLATE=en_US.UTF-8    LC_CTYPE=en_US.UTF-8
[2] LC_MONETARY=en_US.UTF-8   LC_NUMERIC=C
[3] LC_PAPER=en_US.UTF-8      LC_TIME=en_US.UTF-8
[4] LC_ADDRESS=C              LC_TELEPHONE=C
[5] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats graphics grDevices utils datasets methods base
other attached packages:
[1] gridExtra_2.3 SparseM_1.77 Hmisc_4.1-1 ggplot2_3.1.0 Formula_1.2-3
[6] lattice_0.20-38 survival_2.43-3 bujar_0.2-4

loaded via a namespace (and not attached):
[1] viridis_0.5.1  viridisLite_0.3.0   splines_3.5.2  foreach_1.4.4
[5] elasticnet_1.1.1 stats4_3.5.2    latticeExtra_0.6-28 stabs_0.6-3
[9] numDeriv_2016.8-1 pillar_1.3.1    backports_1.1.3  quantreg_5.38
[13] quadprog_1.5-5  mboost_2.9-1     digest_0.6.18   RColorBrewer_1.1-2
[17] checkmate_1.9.0 colorspace_1.3-2 sandwich_2.5-0  gbm_2.1.4
[21] rms_5.1-2 htmltools_0.3.6     Matrix_1.2-15   plyr_1.8.4
[25] pkgconfig_2.0.2 earth_4.7.0    mvtnorm_1.0-8   scales_1.0.0
[29] TeachingDemos_2.10 MatrixModels_0.4-1 htmlTable_1.13.1  tibble_2.0.0
[33] mda_0.4-10 withr_2.1.2       TH.data_1.0-9   nnet_7.3-12
[37] lazyeval_0.2.1 magrittr_1.5     crayon_1.3.4   polspline_1.1.13
[41] doParallel_1.0.14 nlme_3.1-137 MASS_7.3-51.1  foreign_0.8-71
[45] class_7.3-15 tools_3.5.2     data.table_1.11.8 multcomp_1.4-8
[49] stringr_1.3.1 munsell_0.5.0    glmnet_2.0-16    cluster_2.0.7-1
[53] plotrix_3.7-4  compiler_3.5.2   inum_1.0-0     rlang_0.3.1
[57] plotmo_3.5.2  grid_3.5.2      iterators_1.0.10 rstudioapi_0.9.0
[61] mpath_0.3-7  htmlwidgets_1.3   base64enc_0.1-3 partykit_1.2-2
[65] gtable_0.2.0  codetools_0.2-16  lars_1.2        bst_0.3-15
[69] nlst_1.4 zoo_1.8-4       knitr_1.21    libcoin_1.0-2
[73] modeltools_0.2-22 stringi_1.2.4  psci_1.5.2   parallel_3.5.2
[77] Rcpp_1.0.0  rpart_4.1-13    acepack_1.4.1  xfun_0.4
References


Affiliation:

Zhu Wang  
Department of Epidemiology & Biostatistics  
UT Health San Antonio  
E-mail: wangz1@uthscsa.edu