assignment2

2023-02-15

Running the file provided by Professor first and then starting with assignment 02

```
#HW02.R
#run a lasso and elastic net model
library(survival)
library(coxed)
```

```
## Loading required package: rms

## Loading required package: Hmisc

## Loading required package: lattice

## Loading required package: Formula

## Loading required package: ggplot2

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, units

## Loading required package: SparseM

##
## Attaching package: 'SparseM'

## The following object is masked from 'package:base':
##
##     backsolve

## Loading required package: mgcv

## Loading required package: nlme

## This is mgcv 1.8-40. For overview type 'help("mgcv-package")'.
```

```
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-4
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
#Simulate time-to-event data
set.seed(123)
n <- 1000

x1 = matrix(rbinom(n * 10, size = 1, prob = 0.3), n, 10)
```

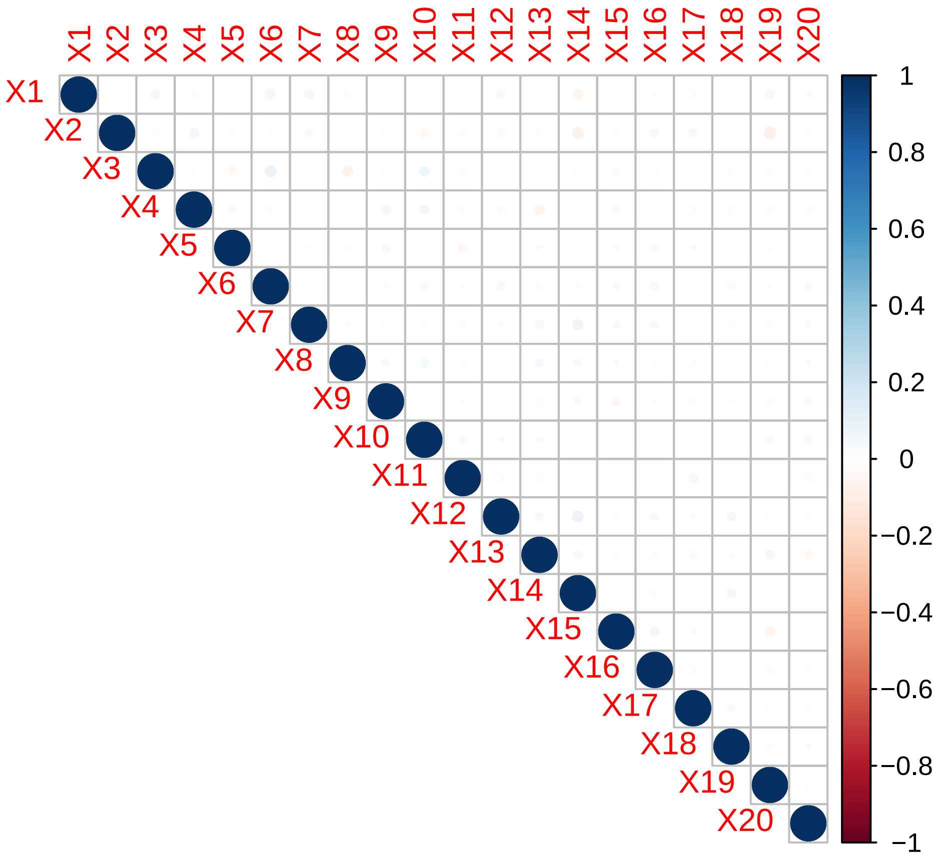
```r
x2 = matrix(rnorm(n * 10,mean=0,sd=.4),n,10)

simdata <- sim.survdata(N=n,T=365, censor=0.15,
                        beta = c(0.4, 0.4, 0.5, 0.5, 0.5, 0, 0, 0, 0, 0,
                                 0.4, 0.4, 0.5, 0.5, 0.5, 0, 0, 0, 0, 0),
                        X = cbind(x1,x2))

data <- simdata$data
attach(data)

setwd("~/StatisticalModelingandComputing/")
#save graphics output in pdf - saves graph(s) in working directory
 pdf(file="HW02_out.pdf")
```

```r
cormat <- round(cor(data[,1:20]),2)
cormat #the correlation coefficients among the predictors are small
```

```
##        X1    X2    X3    X4    X5    X6    X7    X8    X9   X10   X11   X12
## X1   1.00  0.00  0.05 -0.03 -0.01 -0.06  0.05 -0.02  0.00  0.01  0.00 -0.04
## X2   0.00  1.00  0.01  0.05  0.00 -0.01  0.03 -0.01  0.00 -0.04 -0.02 -0.03
## X3   0.05  0.01  1.00 -0.01 -0.04  0.08  0.01 -0.07  0.00  0.06 -0.02  0.00
## X4  -0.03  0.05 -0.01  1.00 -0.04  0.02 -0.01  0.00  0.05  0.05  0.02 -0.03
## X5  -0.01  0.00 -0.04 -0.04  1.00 -0.01 -0.02  0.01  0.04  0.01 -0.05 -0.01
## X6  -0.06 -0.01  0.08  0.02 -0.01  1.00  0.00  0.01  0.02 -0.04  0.01  0.04
## X7   0.05  0.03  0.01 -0.01 -0.02  0.00  1.00  0.02  0.00 -0.01 -0.03  0.02
## X8  -0.02 -0.01 -0.07  0.00  0.01  0.01  0.02  1.00  0.03  0.04  0.01  0.01
## X9   0.00  0.00  0.00  0.05  0.04  0.02  0.00  0.03  1.00  0.01  0.01  0.01
## X10  0.01 -0.04  0.06  0.05  0.01 -0.04 -0.01  0.04  0.01  1.00 -0.04  0.02
## X11  0.00 -0.02 -0.02  0.02 -0.05  0.01 -0.03  0.01  0.01 -0.04  1.00 -0.02
## X12 -0.04 -0.03  0.00 -0.03 -0.01  0.04  0.02  0.01  0.01  0.02 -0.02  1.00
## X13  0.00 -0.02  0.00 -0.06  0.02 -0.02 -0.05  0.04 -0.02  0.02 -0.02  0.04
## X14 -0.06 -0.08  0.00 -0.01 -0.01 -0.02  0.07  0.03  0.03 -0.01  0.00  0.08
## X15  0.00 -0.01 -0.03 -0.04  0.02  0.03  0.03  0.02 -0.05  0.01 -0.01  0.00
## X16  0.01  0.04  0.00 -0.01  0.03 -0.04 -0.04  0.01  0.01  0.00  0.00 -0.04
## X17  0.02  0.04  0.00 -0.02  0.02  0.00  0.00 -0.02  0.02  0.00 -0.05  0.02
## X18 -0.01 -0.01 -0.02  0.02  0.00 -0.03  0.01 -0.01  0.01  0.00  0.01  0.05
## X19 -0.05 -0.09 -0.03 -0.03 -0.02  0.01  0.02  0.00 -0.03  0.03 -0.01  0.00
## X20  0.02  0.00 -0.01 -0.03  0.00  0.03  0.02  0.02 -0.04 -0.04 -0.03  0.00
##       X13   X14   X15   X16   X17   X18   X19   X20
## X1   0.00 -0.06  0.00  0.01  0.02 -0.01 -0.05  0.02
## X2  -0.02 -0.08 -0.01  0.04  0.04 -0.01 -0.09  0.00
## X3   0.00  0.00 -0.03  0.00  0.00 -0.02 -0.03 -0.01
## X4  -0.06 -0.01 -0.04 -0.01 -0.02  0.02 -0.03 -0.03
## X5   0.02 -0.01  0.02  0.03  0.02  0.00 -0.02  0.00
## X6  -0.02 -0.02  0.03 -0.04  0.00 -0.03  0.01  0.03
## X7  -0.05  0.07  0.03 -0.04  0.00  0.01  0.02  0.02
## X8   0.04  0.03  0.02  0.01 -0.02 -0.01  0.00  0.02
## X9  -0.02  0.03 -0.05  0.01  0.02  0.01 -0.03 -0.04
## X10  0.02 -0.01  0.01  0.00  0.00  0.00  0.03 -0.04
## X11 -0.02  0.00 -0.01  0.00 -0.05  0.01 -0.01 -0.03
## X12  0.04  0.08  0.00 -0.04  0.02  0.05  0.00  0.00
## X13  1.00 -0.04  0.01  0.02 -0.03  0.02  0.05 -0.04
## X14 -0.04  1.00 -0.01  0.02  0.01  0.05 -0.01 -0.01
```

```
## X15  0.01 -0.01  1.00  0.05  0.02 -0.01 -0.06 -0.01
## X16  0.02  0.02  0.05  1.00 -0.02  0.00 -0.03  0.00
## X17 -0.03  0.01  0.02 -0.02  1.00  0.03  0.00  0.01
## X18  0.02  0.05 -0.01  0.00  0.03  1.00 -0.02  0.02
## X19  0.05 -0.01 -0.06 -0.03  0.00 -0.02  1.00  0.00
## X20 -0.04 -0.01 -0.01  0.00  0.01  0.02  0.00  1.00
```

```
corrplot(cormat,type="upper")
```




```
model <- coxph(Surv(y, failed) ~ ., data=data)
model$coefficients ## model-estimated coefficients
```

```
##          X1          X2          X3          X4          X5          X6 
##  0.38553508  0.27916537  0.57810539  0.43436549  0.50063956 -0.03073943 
##          X7          X8          X9         X10         X11         X12 
## -0.02313047 -0.01384174  0.09575907  0.06933871  0.42155723  0.28890993 
##         X13         X14         X15         X16         X17         X18 
##  0.55808911  0.40491251  0.29214823 -0.04087630  0.10750969  0.07324933 
##         X19         X20 
## -0.07680564  0.13702470
```

```
summary(model)
```

```
## Call:
## coxph(formula = Surv(y, failed) ~ ., data = data)
## 
##   n= 1000, number of events= 842 
## 
##         coef exp(coef) se(coef)      z Pr(>|z|)
```

```
## X1   0.38554   1.47040  0.07768  4.963 6.94e-07 ***
## X2   0.27917   1.32203  0.07687  3.632 0.000281 ***
## X3   0.57811   1.78266  0.07716  7.492 6.79e-14 ***
## X4   0.43437   1.54398  0.07754  5.602 2.12e-08 ***
## X5   0.50064   1.64978  0.07956  6.293 3.12e-10 ***
## X6  -0.03074   0.96973  0.07680 -0.400 0.688981
## X7  -0.02313   0.97713  0.07762 -0.298 0.765693
## X8  -0.01384   0.98625  0.07679 -0.180 0.856960
## X9   0.09576   1.10049  0.07838  1.222 0.221807
## X10  0.06934   1.07180  0.07638  0.908 0.363981
## X11  0.42156   1.52433  0.09187  4.589 4.46e-06 ***
## X12  0.28891   1.33497  0.08811  3.279 0.001042 **
## X13  0.55809   1.74733  0.09007  6.196 5.80e-10 ***
## X14  0.40491   1.49917  0.08823  4.589 4.45e-06 ***
## X15  0.29215   1.33930  0.09136  3.198 0.001384 **
## X16 -0.04088   0.95995  0.08892 -0.460 0.645749
## X17  0.10751   1.11350  0.08645  1.244 0.213623
## X18  0.07325   1.07600  0.08728  0.839 0.401355
## X19 -0.07681   0.92607  0.09030 -0.851 0.395021
## X20  0.13702   1.14686  0.08709  1.573 0.115613
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##     exp(coef) exp(-coef) lower .95 upper .95
## X1     1.4704     0.6801    1.2627     1.712
## X2     1.3220     0.7564    1.1371     1.537
## X3     1.7827     0.5610    1.5324     2.074
## X4     1.5440     0.6477    1.3263     1.797
## X5     1.6498     0.6061    1.4116     1.928
## X6     0.9697     1.0312    0.8342     1.127
## X7     0.9771     1.0234    0.8392     1.138
## X8     0.9863     1.0139    0.8484     1.146
## X9     1.1005     0.9087    0.9438     1.283
## X10    1.0718     0.9330    0.9228     1.245
## X11    1.5243     0.6560    1.2732     1.825
## X12    1.3350     0.7491    1.1232     1.587
## X13    1.7473     0.5723    1.4645     2.085
## X14    1.4992     0.6670    1.2611     1.782
## X15    1.3393     0.7467    1.1197     1.602
## X16    0.9599     1.0417    0.8064     1.143
## X17    1.1135     0.8981    0.9400     1.319
## X18    1.0760     0.9294    0.9068     1.277
## X19    0.9261     1.0798    0.7759     1.105
## X20    1.1469     0.8719    0.9669     1.360
##
## Concordance= 0.644  (se = 0.011 )
## Likelihood ratio test= 208.6  on 20 df,   p=<2e-16
## Wald test            = 208.8  on 20 df,   p=<2e-16
## Score (logrank) test = 211.3  on 20 df,   p=<2e-16
```

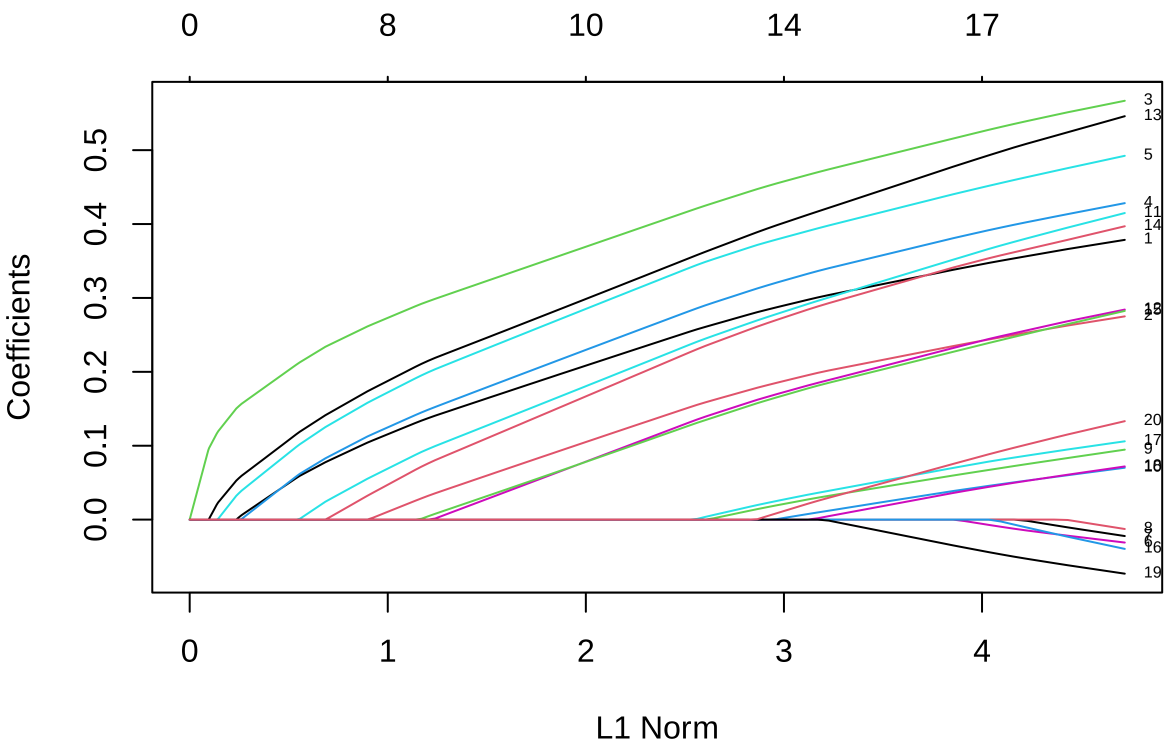
```
# Perform survival analysis using glmnet


# Perform survival analysis using glmnet
```

```
# assuming the predictors are in the first p columns and the response is in the last two columns
p <- ncol(data) - 2
p
```

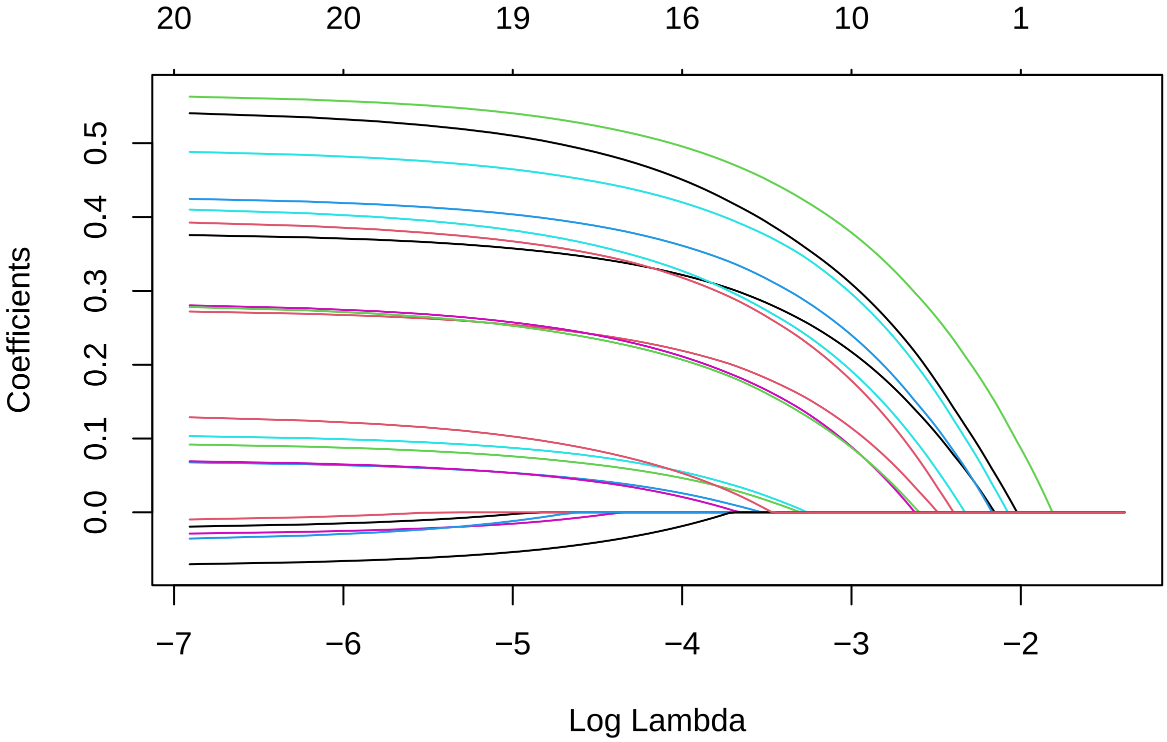
```
## [1] 20
```

```
xmatrix=as.matrix(data[,1:p])
show_fit=glmnet(xmatrix,Surv(y, failed),standardize=TRUE,lambda=seq(0,0.25,.001),alpha=1,family = "cox")
#print(show_fit)

plot(show_fit,label=TRUE)
```

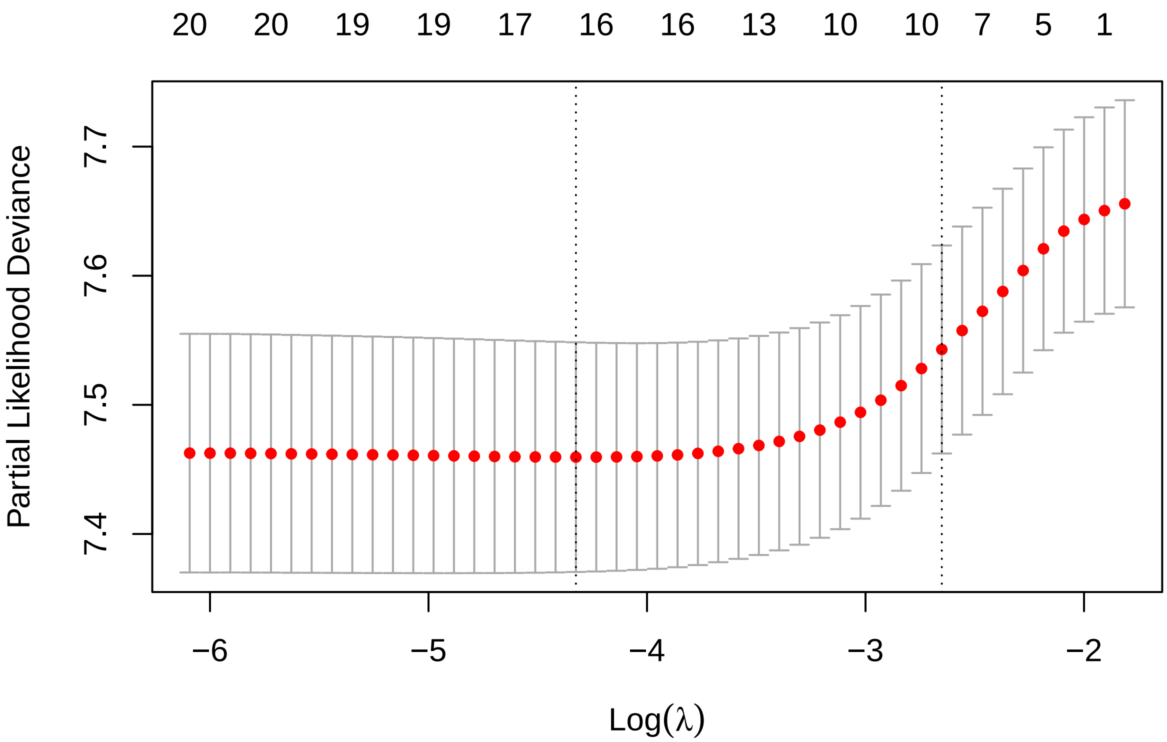
```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```




```
plot(show_fit,xvar = "lambda",label=TRUE)
```




```
#show the use of the CV metric "Partial Likelihood Deviance"
surv_model <- cv.glmnet(xmatrix,Surv(y, failed), family = "cox", type.measure = "deviance",alpha=1,nfol
plot(surv_model)
```




```
print(surv_model)
```

```
##
```

```
## Call:  cv.glmnet(x = xmatrix, y = Surv(y, failed), type.measure = "deviance",      nfolds = 10, famil
##
## Measure: Partial Likelihood Deviance
##
##       Lambda Index Measure      SE Nonzero
## min 0.01323    28   7.460 0.08895      16
## 1se 0.07060    10   7.543 0.08055      10
```

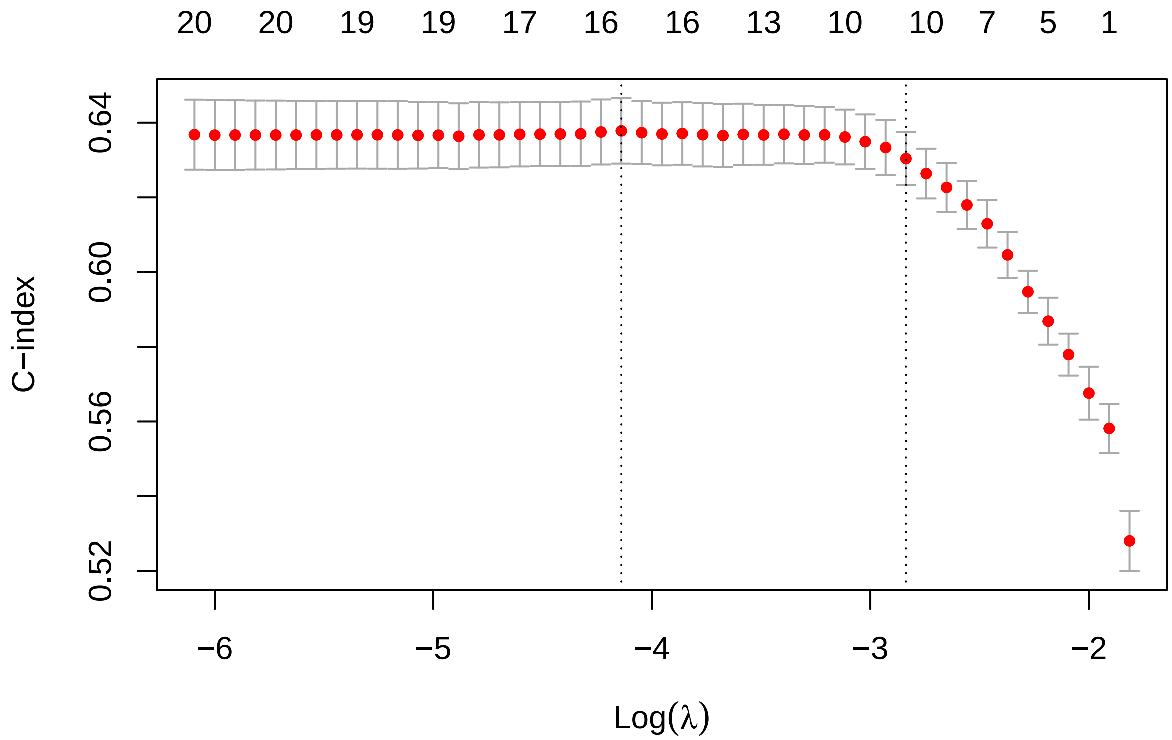
```
# Extract the optimal lambda value at lambda.1se
lambda_opt <- surv_model$lambda.1se

# Fit the model using the optimal lambda value
fit <- glmnet(xmatrix,Surv(y, failed), family = "cox", type.measure = "deviance",alpha=1,lambda = lambda
coef(fit)
```

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##               s0
## X1  0.145632493
## X2  0.040743048
## X3  0.303992347
## X4  0.158165765
## X5  0.209541491
## X6  .
## X7  .
## X8  .
## X9  .
## X10 .
## X11 0.105349378
## X12 0.007016571
## X13 0.224756848
## X14 0.086701559
## X15 0.012951787
## X16 .
## X17 .
## X18 .
## X19 .
## X20 .
```

```
# Make predictions
predictions <- predict(fit, newx = xmatrix[,1:p], type = "response")

# Evaluate the model performance using concordance index (c-index)
c_index <- Cindex(predictions,Surv(y, failed))
c_index
```

```
## [1] 0.6351173
```

```
#show the use of CV metric "C"
surv_model_C = cv.glmnet(xmatrix,Surv(y, failed), family = "cox", type.measure = "C", alpha=1,nfolds = 
plot(surv_model_C)
```




```
print(surv_model_C)
```

```
##
## Call:  cv.glmnet(x = xmatrix, y = Surv(y, failed), type.measure = "C",      nfolds = 10, family = "co
##
## Measure: C-index
##
##      Lambda Index Measure       SE Nonzero
## min 0.01593    26  0.6378 0.008766      16
## 1se 0.05861    12  0.6304 0.007090      10
```

```
# Extract the optimal lambda value at lambda.1se
lambda_opt <- surv_model_C$lambda.1se

# Fit the model using the optimal lambda value
fit_C <- glmnet(xmatrix,Surv(y, failed), family = "cox", type.measure = "C",alpha=1,lambda = lambda_opt)
coef(fit_C)
```

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##             s0
## X1  0.18707001
## X2  0.08315443
## X3  0.34697746
## X4  0.20533298
## X5  0.25941205
## X6  .
## X7  .
## X8  .
## X9  .
## X10 .
## X11 0.15478563
```

```
## X12 0.05398264
## X13 0.27324345
## X14 0.13936475
## X15 0.05567932
## X16 .
## X17 .
## X18 .
## X19 .
## X20 .
```

```
# Make predictions
predictions <- predict(fit_C, newx = xmatrix[,1:p], type = "response")
# Evaluate the model performance using concordance index (c-index)
c_index <- Cindex(predictions,Surv(y, failed))
c_index
```

```
## [1] 0.6399806
```

Assignment 2

a. Redo the simulation where you select 5 of the X coefficients among the binary X's to be 0 and 5 of the X coefficients among the normally distributed X's to be 0.

Note: just modify the line in the program: beta = c(0.4, 0.4, 0.5, 0.5, 0.5, 0, 0, 0, 0, 0, 0.4, 0.4, 0.5, 0.5, 0.5, 0, 0, 0, 0, 0),

Generate the text output and the graphical output to answer parts b - f below.

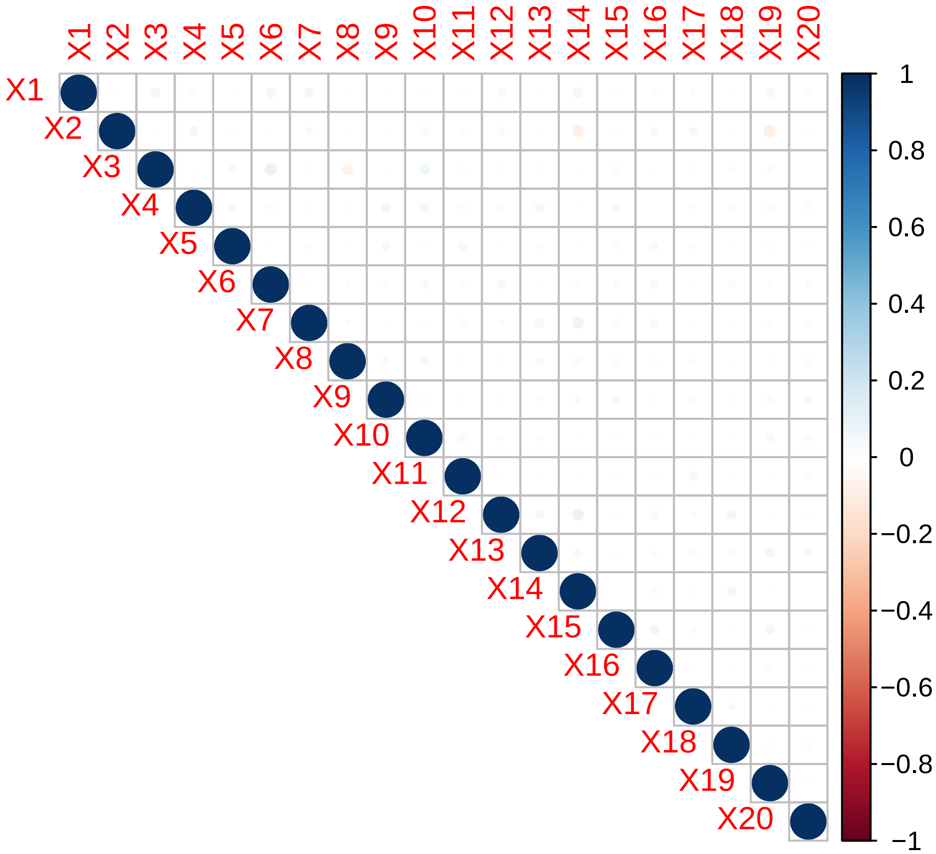
```
#Simulate time-to-event data
set.seed(123)
n <- 1000

x1 = matrix(rbinom(n * 10, size = 1, prob = 0.3), n, 10)

x2 = matrix(rnorm(n * 10,mean=0,sd=.4),n,10)

simdata <- sim.survdata(N=n,T=365, censor=0.15,
                        beta = c(0, 0.4, 0.5, 0, 0.5, 0, 0.4, 0.4, 0, 0.5,
                                 0.5, 0, 0, 0, 0.4, 0, 0.5, 0.5, 0, 0),
                        X = cbind(x1,x2))

data <- simdata$data
```

```
cormat <- round(cor(data[,1:20]),2)
cormat #the correlation coefficients among the predictors are small
```

```
##        X1    X2    X3    X4    X5    X6    X7    X8    X9   X10   X11   X12
## X1   1.00  0.00  0.05 -0.03 -0.01 -0.06  0.05 -0.02  0.00  0.01  0.00 -0.04
## X2   0.00  1.00  0.01  0.05  0.00 -0.01  0.03 -0.01  0.00 -0.04 -0.02 -0.03
## X3   0.05  0.01  1.00 -0.01 -0.04  0.08  0.01 -0.07  0.00  0.06 -0.02  0.00
## X4  -0.03  0.05 -0.01  1.00 -0.04  0.02 -0.01  0.00  0.05  0.05  0.02 -0.03
## X5  -0.01  0.00 -0.04 -0.04  1.00 -0.01 -0.02  0.01  0.04  0.01 -0.05 -0.01
## X6  -0.06 -0.01  0.08  0.02 -0.01  1.00  0.00  0.01  0.02 -0.04  0.01  0.04
## X7   0.05  0.03  0.01 -0.01 -0.02  0.00  1.00  0.02  0.00 -0.01 -0.03  0.02
## X8  -0.02 -0.01 -0.07  0.00  0.01  0.01  0.02  1.00  0.03  0.04  0.01  0.01
## X9   0.00  0.00  0.00  0.05  0.04  0.02  0.00  0.03  1.00  0.01  0.01  0.01
## X10  0.01 -0.04  0.06  0.05  0.01 -0.04 -0.01  0.04  0.01  1.00 -0.04  0.02
## X11  0.00 -0.02 -0.02  0.02 -0.05  0.01 -0.03  0.01  0.01 -0.04  1.00 -0.02
```

```
## X12 -0.04 -0.03  0.00 -0.03 -0.01  0.04  0.02  0.01  0.01  0.02 -0.02  1.00
## X13  0.00 -0.02  0.00 -0.06  0.02 -0.02 -0.05  0.04 -0.02  0.02 -0.02  0.04
## X14 -0.06 -0.08  0.00 -0.01 -0.01 -0.02  0.07  0.03  0.03 -0.01  0.00  0.08
## X15  0.00 -0.01 -0.03 -0.04  0.02  0.03  0.03  0.02 -0.05  0.01 -0.01  0.00
## X16  0.01  0.04  0.00 -0.01  0.03 -0.04 -0.04  0.01  0.01  0.00  0.00 -0.04
## X17  0.02  0.04  0.00 -0.02  0.02  0.00  0.00 -0.02  0.02  0.00 -0.05  0.02
## X18 -0.01 -0.01 -0.02  0.02  0.00 -0.03  0.01 -0.01  0.01  0.00  0.01  0.05
## X19 -0.05 -0.09 -0.03 -0.03 -0.02  0.01  0.02  0.00 -0.03  0.03 -0.01  0.00
## X20  0.02  0.00 -0.01 -0.03  0.00  0.03  0.02  0.02 -0.04 -0.04 -0.03  0.00
##       X13   X14   X15   X16   X17   X18   X19   X20
## X1   0.00 -0.06  0.00  0.01  0.02 -0.01 -0.05  0.02
## X2  -0.02 -0.08 -0.01  0.04  0.04 -0.01 -0.09  0.00
## X3   0.00  0.00 -0.03  0.00  0.00 -0.02 -0.03 -0.01
## X4  -0.06 -0.01 -0.04 -0.01 -0.02  0.02 -0.03 -0.03
## X5   0.02 -0.01  0.02  0.03  0.02  0.00 -0.02  0.00
## X6  -0.02 -0.02  0.03 -0.04  0.00 -0.03  0.01  0.03
## X7  -0.05  0.07  0.03 -0.04  0.00  0.01  0.02  0.02
## X8   0.04  0.03  0.02  0.01 -0.02 -0.01  0.00  0.02
## X9  -0.02  0.03 -0.05  0.01  0.02  0.01 -0.03 -0.04
## X10  0.02 -0.01  0.01  0.00  0.00  0.00  0.03 -0.04
## X11 -0.02  0.00 -0.01  0.00 -0.05  0.01 -0.01 -0.03
## X12  0.04  0.08  0.00 -0.04  0.02  0.05  0.00  0.00
## X13  1.00 -0.04  0.01  0.02 -0.03  0.02  0.05 -0.04
## X14 -0.04  1.00 -0.01  0.02  0.01  0.05 -0.01 -0.01
## X15  0.01 -0.01  1.00  0.05  0.02 -0.01 -0.06 -0.01
## X16  0.02  0.02  0.05  1.00 -0.02  0.00 -0.03  0.00
## X17 -0.03  0.01  0.02 -0.02  1.00  0.03  0.00  0.01
## X18  0.02  0.05 -0.01  0.00  0.03  1.00 -0.02  0.02
## X19  0.05 -0.01 -0.06 -0.03  0.00 -0.02  1.00  0.00
## X20 -0.04 -0.01 -0.01  0.00  0.01  0.02  0.00  1.00
```

```r
corrplot(cormat,type="upper")
```




```r
model <- coxph(Surv(y, failed) ~ ., data=data)
model$coefficients ## model-estimated coefficients
```

```
##          X1          X2          X3          X4          X5          X6 
## -0.02761794  0.28551803  0.58264648 -0.06756285  0.49590014 -0.04703972 
##          X7          X8          X9         X10         X11         X12 
##  0.38352880  0.40301909  0.08843731  0.56915107  0.51425466 -0.12223634 
##         X13         X14         X15         X16         X17         X18 
##  0.04293680 -0.10845914  0.19784765 -0.03132104  0.62660513  0.58952345 
##         X19         X20 
## -0.07229006  0.14748694
```

```r
summary(model)
```

```
## Call:
## coxph(formula = Surv(y, failed) ~ ., data = data)
## 
##   n= 1000, number of events= 842 
## 
##         coef exp(coef) se(coef)      z Pr(>|z|)    
## X1  -0.02762   0.97276  0.07672 -0.360 0.718849    
## X2   0.28552   1.33045  0.07694  3.711 0.000206 ***
## X3   0.58265   1.79077  0.07714  7.553 4.27e-14 ***
## X4  -0.06756   0.93467  0.07629 -0.886 0.375857    
## X5   0.49590   1.64198  0.07956  6.233 4.58e-10 ***
## X6  -0.04704   0.95405  0.07709 -0.610 0.541741    
## X7   0.38353   1.46745  0.07834  4.896 9.80e-07 ***
## X8   0.40302   1.49634  0.07767  5.189 2.11e-07 ***
```

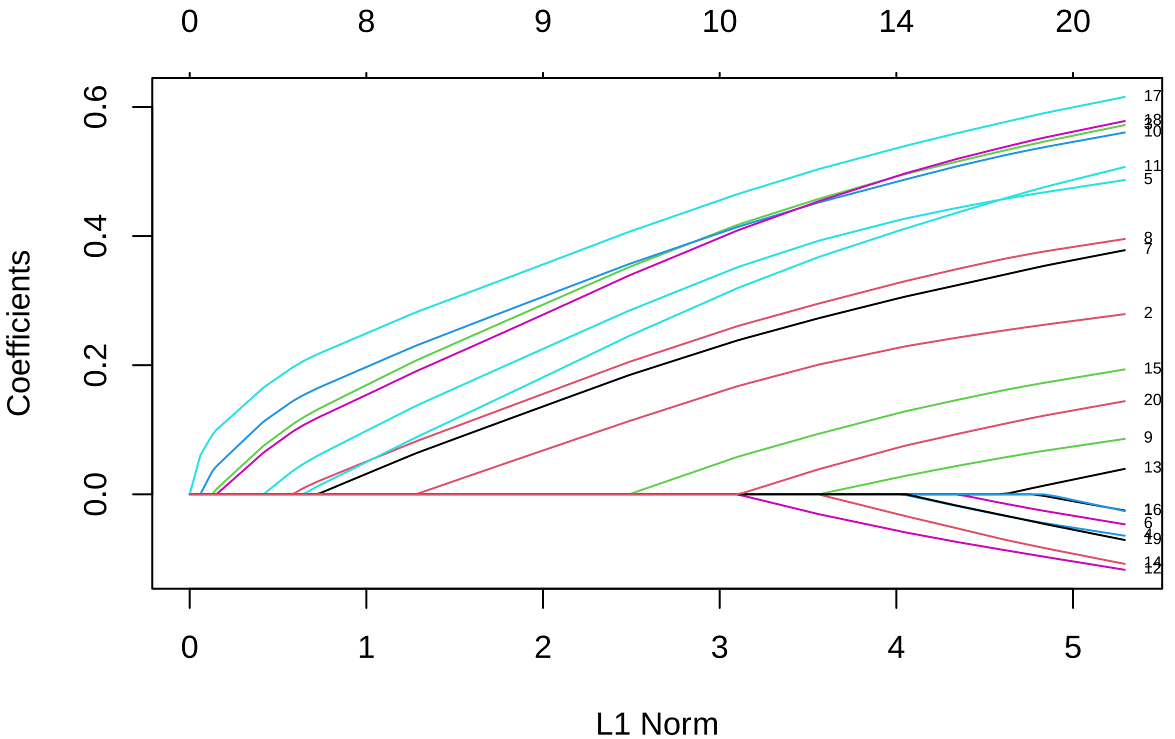
```
## X9   0.08844   1.09247  0.07836  1.129 0.259037
## X10  0.56915   1.76677  0.07801  7.296 2.97e-13 ***
## X11  0.51425   1.67239  0.09230  5.572 2.52e-08 ***
## X12 -0.12224   0.88494  0.08817 -1.386 0.165635
## X13  0.04294   1.04387  0.08778  0.489 0.624736
## X14 -0.10846   0.89722  0.08734 -1.242 0.214301
## X15  0.19785   1.21878  0.09071  2.181 0.029178 *
## X16 -0.03132   0.96916  0.08918 -0.351 0.725427
## X17  0.62661   1.87125  0.08843  7.086 1.38e-12 ***
## X18  0.58952   1.80313  0.08929  6.603 4.04e-11 ***
## X19 -0.07229   0.93026  0.09038 -0.800 0.423807
## X20  0.14749   1.15892  0.08721  1.691 0.090807 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##     exp(coef) exp(-coef) lower .95 upper .95
## X1     0.9728     1.0280    0.8370     1.131
## X2     1.3305     0.7516    1.1442     1.547
## X3     1.7908     0.5584    1.5395     2.083
## X4     0.9347     1.0699    0.8049     1.085
## X5     1.6420     0.6090    1.4049     1.919
## X6     0.9540     1.0482    0.8203     1.110
## X7     1.4675     0.6815    1.2586     1.711
## X8     1.4963     0.6683    1.2851     1.742
## X9     1.0925     0.9154    0.9369     1.274
## X10    1.7668     0.5660    1.5163     2.059
## X11    1.6724     0.5979    1.3956     2.004
## X12    0.8849     1.1300    0.7445     1.052
## X13    1.0439     0.9580    0.8789     1.240
## X14    0.8972     1.1146    0.7561     1.065
## X15    1.2188     0.8205    1.0203     1.456
## X16    0.9692     1.0318    0.8137     1.154
## X17    1.8712     0.5344    1.5735     2.225
## X18    1.8031     0.5546    1.5137     2.148
## X19    0.9303     1.0750    0.7792     1.111
## X20    1.1589     0.8629    0.9768     1.375
##
## Concordance= 0.676  (se = 0.01 )
## Likelihood ratio test= 281.2  on 20 df,   p=<2e-16
## Wald test            = 284.6  on 20 df,   p=<2e-16
## Score (logrank) test = 287.7  on 20 df,   p=<2e-16
```

```r
# Perform survival analysis using glmnet


# Perform survival analysis using glmnet
# assuming the predictors are in the first p columns and the response is in the last two columns
p <- ncol(data) - 2
xmatrix=as.matrix(data[,1:p])
show_fit=glmnet(xmatrix,Surv(data$y, data$failed),standardize=TRUE,lambda=seq(0,0.25,.001),alpha=1,famil
#print(show_fit)
```

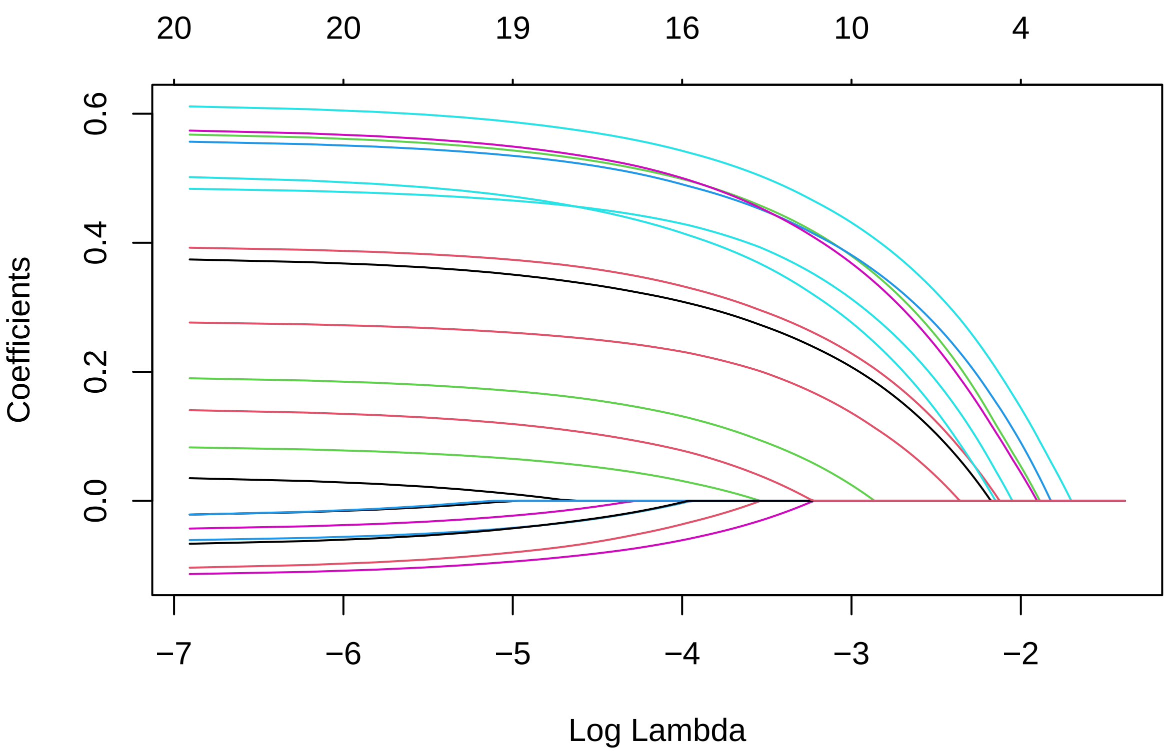
```r
plot(show_fit,label=TRUE)
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
```

```
## collapsing to unique 'x' values
```

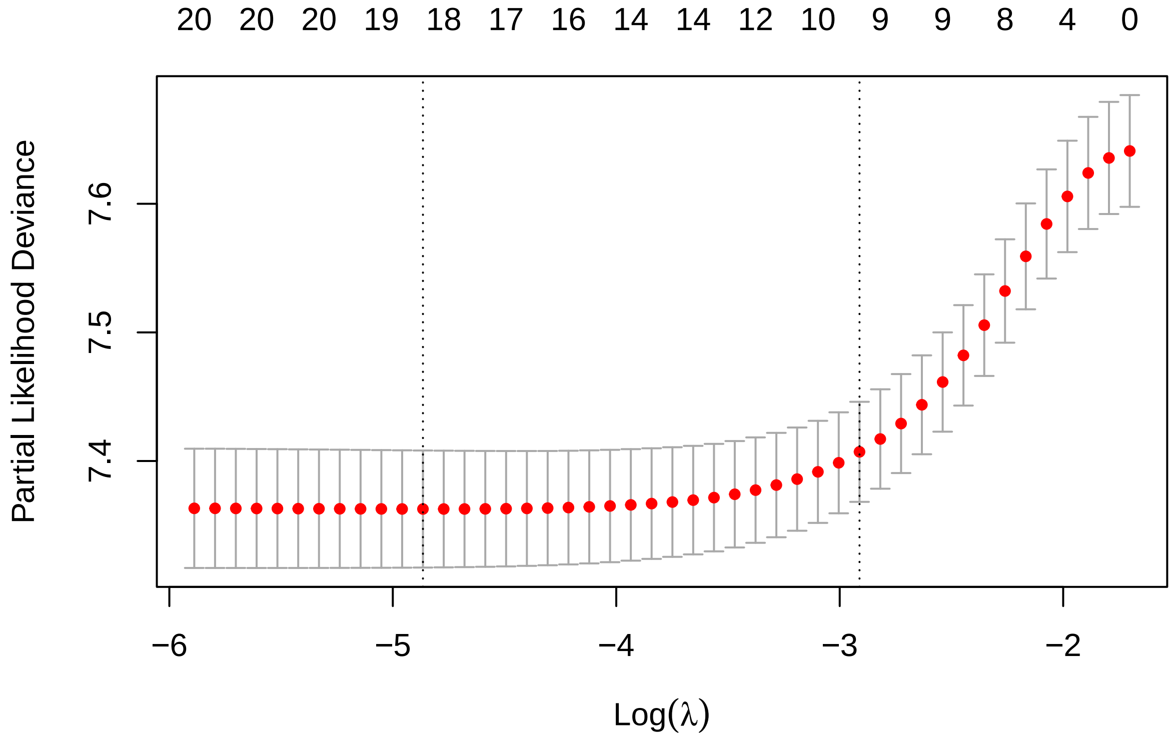



```
plot(show_fit,xvar = "lambda",label=TRUE)
```



b. Did the Lasso proportional hazards model correctly select the 10 significant predictors using the partial likelihood deviance metric? Comment.

```r
#show the use of the CV metric "Partial Likelihood Deviance"
surv_model <- cv.glmnet(xmatrix,Surv(data$y, data$failed), family = "cox", type.measure = "deviance",al
plot(surv_model)
```




```r
print(surv_model)
```

```
##
## Call:  cv.glmnet(x = xmatrix, y = Surv(data$y, data$failed), type.measure = "deviance",      nfolds =
##
## Measure: Partial Likelihood Deviance
##
##      Lambda Index Measure     SE Nonzero
## min 0.00771    35   7.363 0.0455      18
## 1se 0.05440    14   7.407 0.0389      10
```

```r
# Extract the optimal lambda value at lambda.1se
lambda_opt <- surv_model$lambda.1se

# Fit the model using the optimal lambda value
fit <- glmnet(xmatrix,Surv(data$y, data$failed), family = "cox", type.measure = "deviance",alpha=1,lamb
coef(fit)
```

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##              s0
## X1  .
## X2  0.121750947
## X3  0.362140703
## X4  .
## X5  0.294950075
## X6  .
## X7  0.192964532
```

```
## X8  0.213611383
## X9  .
## X10 0.365617547
## X11 0.256698713
## X12 .
## X13 .
## X14 .
## X15 0.008988588
## X16 .
## X17 0.415643416
## X18 0.349694016
## X19 .
## X20 .
```

All the 10 significant predictors were selected by using partial likelihood deviance without penalizing any coefficient to 0.

c. What is the C-index for the optimal lambda chosen to be Lambda.1se using the partial likelihood deviance metric?

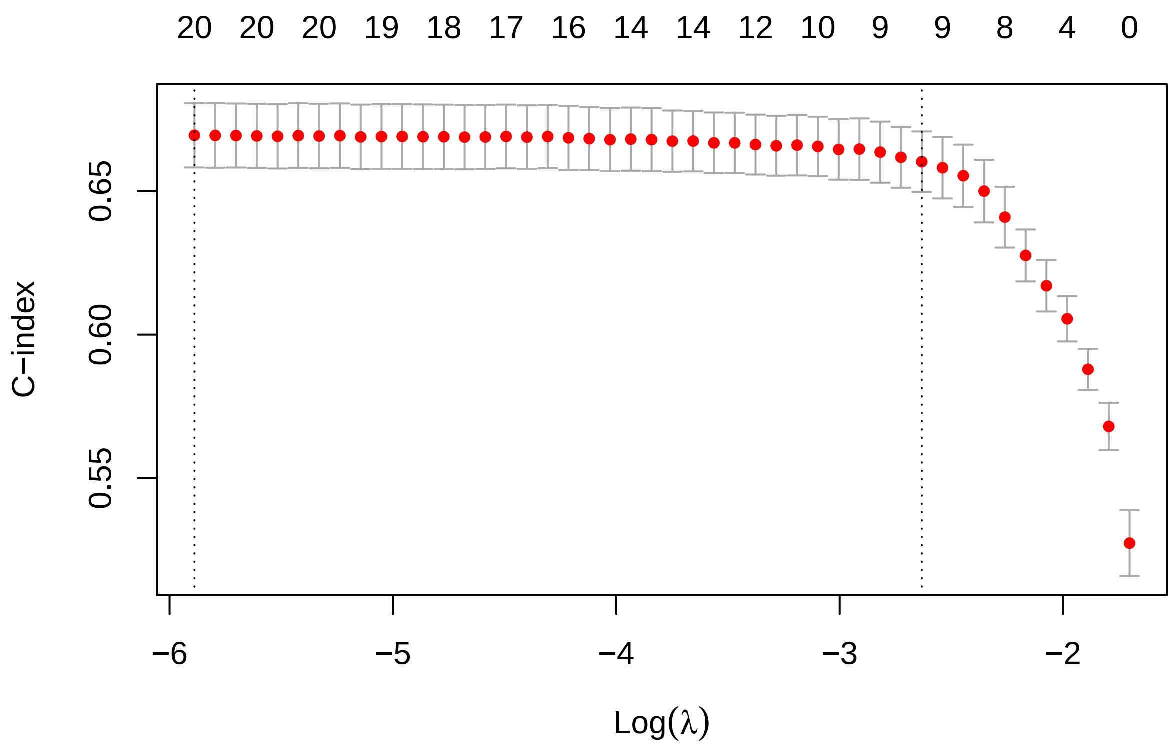
```r
# Make predictions
predictions <- predict(fit, newx = xmatrix[,1:p], type = "response")

# Evaluate the model performance using concordance index (c-index)
c_index <- Cindex(predictions,Surv(data$y, data$failed))
c_index
```

```
## [1] 0.6697602
```

The C-index using likelihood deviance metric is 0.6697602

d. Did the Lasso proportional hazards model correctly select the 10 significant predictors using Cross Validation with the C-index metric? Comment.

```r
#show the use of CV metric "C"
surv_model_C = cv.glmnet(xmatrix,Surv(data$y, data$failed), family = "cox", type.measure = "C", alpha=1
plot(surv_model_C)
```




```
print(surv_model_C)
```

```
##
## Call:  cv.glmnet(x = xmatrix, y = Surv(data$y, data$failed), type.measure = "C",      nfolds = 10, fa
##
## Measure: C-index
##
##      Lambda Index Measure      SE Nonzero
## min 0.00277    46  0.6694 0.01120      20
## 1se 0.07191    11  0.6602 0.01054       9
```

```
# Extract the optimal lambda value at lambda.1se
lambda_opt <- surv_model_C$lambda.1se

# Fit the model using the optimal lambda value
fit_C <- glmnet(xmatrix,Surv(data$y, data$failed), family = "cox", type.measure = "C",alpha=1,lambda = l
coef(fit_C)
```

```
## 20 x 1 sparse Matrix of class "dgCMatrix"
##             s0
## X1   .
## X2  0.06869792
## X3  0.29469540
## X4   .
## X5  0.22618656
## X6   .
## X7  0.13685449
## X8  0.15623710
## X9   .
## X10 0.30678742
## X11 0.18229053
```

```
## X12 .
## X13 .
## X14 .
## X15 .
## X16 .
## X17 0.35684786
## X18 0.27916891
## X19 .
## X20 .
```

Using CV all 10 significant predictors were selected without penalizing any coefficient to 0.

e. What is the C-index for the optimal lambda chosen to be Lambda.1se using the C-index metric?

```r
# Make predictions
predictions <- predict(fit_C, newx = xmatrix[,1:p], type = "response")
# Evaluate the model performance using concordance index (c-index)
c_index <- Cindex(predictions,Surv(data$y, data$failed))
c_index
```

```
## [1] 0.6673913
```

The C-index using C-index metric is 0.6673913

f. Comment using just 1 sentence on the size of the correlations between pairs of the predictors.

Correlation between the pairs of predictors is small as its smaller than 1 and not very different than 0.6

```r
detach(data)
##------------------------------------------------------------------##
```