Package 'gboost'

March 30, 2015

Type Package

Title High-dimensional Variable Selection in Survival Data via Gradient Boosting

Version 1.0.1

Date 2015-03-30

Author Lu Xia, Zhi (Kevin) He

Maintainer Lu Xia <luxia@umich.edu>

Description This package implements a gradient boosting algorithm to perform automated variable selection in survival data analysis. It returns the coefficient estimates, indices of selected variables, optimal log partial likelihood and model size. Cross-validation can be enabled.

License GPL-2

R topics documented:

gboost-package *High-dimensional Variable Selection in Survival Data via Gradient Boosting*

Description

This package implements a gradient boosting algorithm to perform automated variable selection in survival data analysis. It returns the coefficient estimates, indices of selected variables, optimal log partial likelihood and model size. Cross-validation can be enabled.

Details

Package:	gboost
Type:	Package
Version:	1.0.1
Date:	2015-03-30
License:	GPL-2

Survival time, censoring indicator and covariates can be passed to the main functions as either vectors and matrices, or as data frames. The package includes four main functions:

`cv.gboost.default`

`cv.gboost`

`gboost.default`

`gboost`

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

Examples

```
require(mvtnorm)
N <- 600
p <- 50
true.beta <- array(0,p)
for(i in 1:5) { true.beta[(i-1)*10+1] <- (-1)^i*0.3*i}

rho <- 0.1
sigma_tmp <- matrix(0, ncol=10, nrow=10)
sigma_z <- matrix(0, ncol=p, nrow=p)
for(i in 1:10)
  for(j in 1:10)
  {
```

```
    sigma_tmp[i,j] <- rho^{abs(i-j)}
  }
for(i in 1:5)
{
  sigma_z[(1+(i-1)*10):(i*10),(1+(i-1)*10):(i*10)] <- sigma_tmp
}

z <- rmvnorm(N, mean=rep(0,p), sigma=sigma_z)
U <- runif(N, 0, 1)
pre_time <- -log(U)/(1*exp(-0.3*z[,1]+0.6*z[,11]-0.9*z[,21]+1.2*z[,31]-1.5*z[,41]))
pre_censoring <- runif(N, 1, 30)
pre_censoring <- pre_censoring*(pre_censoring<3)+3*(pre_censoring>=3)
delta <- (pre_censoring>=pre_time)
time <- pre_time*(delta==1)+pre_censoring*(delta==0)

## Without formula ##
obj <- cv.gboost.default(time=time, delta=delta, z=z,
                                  nfold=5, track=3, rate=0.01, tol=1.0e-4)
cbind(obj$select.var, round(obj$coefficients[obj$select.var],3))

## With formula ##
test.data <- as.data.frame(cbind(z, time, delta))
names(test.data[,1:50]) <- paste("V", 1:50, sep="")
obj1 <- gboost(formula=Surv1(time, delta)~., data=test.data, cross.validation=TRUE)
summary(obj1)
obj2 <- cv.gboost(formula=Surv1(time, delta)~ V1+ V2+ V11 + V12 + V21 +
                    V22 + V31 + V32 + V41 + V42, data=test.data)
summary(obj2)
plot(obj2)
```

`cv.gboost` *Function to implement cross-validation with data frame and formula as input*

Description

With survival time, censoring indicators and covariates input in a data frame, `cv.gboost` performs k fold cross validation to tune the parameter of updating steps.

Usage

```
cv.gboost(formula, data, nfold = 5, track = 5, rate = 0.01, tol = 1e-04)
```

Arguments

`formula`	specifies variables in the data frame to be responses or covariates. The responses, i.e. observed survival time and censoring indicators are specified by `Surv1(,)` with the first argument being the name of survival time and the second argument being the name of censoring indicators. Specifying covariate part is the same as in `lm()` for linear regression models. See the example for more details.
`data`	a data frame containing observed survival time, censoring indicators and covariates to select from.

nfold	the number of folds in cross validation; default is 5.
track	the number of backtracking steps to examine cross validation risks; default is 5.
rate	a parameter specifying the learning rate; default is 0.01 to achieve slow learning.
tol	a parameter specifying the tolerance level in fitting tilde_beta_j; default is 1.0e-4.

Details

The input data should not contain missing values. Returns an object from class "cv.gboost", to which `summary()` and `plot()` can be applied to get selected variables and their estimated coefficients, model size and optimal number of updating steps, and plot the cross validation risks versus number of updating steps.

Value

Returned values can be directly accessed as in `cv.gboost.default`, including `formula`, `coefficients`, `select.var`, `cv.risk`, `loglik`, `model.size`, `cv.m`. `summary()` is recommended for displaying the fitted results.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

`formula`, `cv.gboost.default`, `plot`, `summary`.

Examples

```
## With formula ##
# test.data <- as.data.frame(cbind(z, time, delta))
# names(test.data[,1:50]) <- paste("V", 1:50, sep="")
# obj2 <- cv.gboost(formula=Surv1(time, delta)~ V1+ V2+ V11 + V12 + V21 +
#                     V22 + V31 + V32 + V41 + V42, data=test.data)
# summary(obj2)
# plot(obj2)
# obj3 <- cv.gboost(formula=Surv1(time, delta)~ ., data=test.data)
```

`cv.gboost.default` *Function to implement cross-validation with input covariate matrices*

Description

With survival time, censoring indicators and covariates input as vectors or matrices, `cv.gboost.default` performs k fold cross validation to tune the parameter of updating steps.

Usage

```
cv.gboost.default(time, delta, z, nfold = 5, track = 5, rate = 0.01, tol = 1e-04)
```

Arguments

`time`	a numerical vector of observed survival time.
`delta`	a numerical vector of censoring indicators. 0 stands for censored and 1 for observed event.
`z`	a numeric matrix of observed covariates, with columns corresponding to covariates and rows to subjects.
`nfold`	the number of folds in cross validation; default is 5.
`track`	the number of backtracking steps to examine cross validation risks; default is 5.
`rate`	a parameter specifying the learning rate; default is 0.01 to achieve slow learning.
`tol`	a parameter specifying the tolerance level in fitting tilde_beta_j; default is 1.0e-4.

Details

The input data should not contain missing values. Returns a list.

Value

`coefficients`	a vector of estimated coefficients corresponding to all covariates in `z`.
`loglik`	the optimal log partial likelihood.
`model.size`	the number of covariates that have non-zero coefficient estimates.
`select.var`	the vector of indices of selected variables, i.e. covariates with non-zero coefficient estimates.
`cv.risk`	the vector of cross validation risks at each iteration step.
`cv.m`	the number of updating steps giving the optimal risk

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

cv.gboost, gboost.default and gboost.

Examples

```
# obj <- cv.gboost.default(time=time, delta=delta, z=z,
#                                  nfold=5, track=3, rate=0.01, tol=1.0e-4)
# cbind(obj$select.var, round(obj$coefficients[obj$select.var],3))
```

---

gboost *Function to implement proposed gradient boosting algorithm with data frame input*

Description

With survival time, censoring indicators and covariates input as a data frame, gboost.default performs variable selection using the proposed gradient boosting algorithm. The option of k fold cross validation to tune the parameter of updating steps can be enabled (default).

Usage

```
gboost(formula, data, cross.validation = TRUE, nfold = 5, track = 5, m.stop = 500, rate = 0.01,
```

Arguments

formula	specifies variables in the data frame to be responses or covariates. The responses, i.e. observed survival time and censoring indicators are specified by Survl(,) with the first argument being the name of survival time and the second argument being the name of censoring indicators. Specifying covariate part is the same as in lm() for linear regression models. See the example for more details.
data	a data frame containing observed survival time, censoring indicators and covariates to select from.
cross.validation	a logical value for whether cross validation is enabled or disabled; default is TRUE.
nfold	the number of folds in cross validation; default is 5.
track	the number of backtracking steps to examine cross validation risks; default is 5.
m.stop	pre-specified number of updating steps. Only useful when cross.validation=FALSE.
rate	a parameter specifying the learning rate; default is 0.01 to achieve slow learning.
tol	a parameter specifying the tolerance level in fitting tilde_beta_j; default is 1.0e-4.

Details

The input data should not contain missing values. Returns an object of class "gboost".

Value

coefficients	a vector of estimated coefficients with each element corresponding to a covariate
likelihood	the estimated maximum log-likelihood.
model.size	the number of covariates that have non-zero coefficients.
select.var	the vector of indices of selected variables.
m.stop	the number of updating steps, as pre-specified if cross.validation=FALSE, or giving the optimal risk if cross.validation=TRUE

.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

summary, formula, gboost.default, cv.gboost.

Examples

```
# test.data <- as.data.frame(cbind(z, time, delta))
# names(test.data[,1:50]) <- paste("V", 1:50, sep="")
# obj1 <- gboost(formula=Surv1(time, delta)~., data=test.data, cross.validation=T)
# summary(obj1)
```

---

gboost.default *Function to implement proposed gradient boosting algorithm with covariate matrices*

Description

With survival time, censoring indicators and covariates input as vectors and matrices, gboost.default performs variable selection using the proposed gradient boosting algorithm. The option of k fold cross validation to tune the parameter of updating steps can be enabled (default).

Usage

```
gboost.default(time, delta, z, cross.validation = TRUE, nfold = 5, track = 5, m.stop = 500, tol
```

Arguments

time	a numerical vector of observed survival time.
delta	a numerical vector of censoring indicators. 0 stands for censored and 1 for observed event.
z	a numeric matrix of observed covariates, with columns corresponding to covariates and rows to subjects.
cross.validation	a logical value for whether cross validation is enabled or disabled; default is TRUE.
nfold	the number of folds in cross validation; default is 5. Only valid when cross.validation=TRUE.
track	the number of backtracking steps to examine cross validation risks; default is 5. Only valid when cross.validation=TRUE.
m.stop	a pre-specified number of updating steps; only useful when cross.validation=FALSE.
tol	a parameter specifying the tolerance level in fitting tilde_beta_j; default is 1.0e-4.
rate	a parameter specifying the learning rate; default is 0.01 to achieve slow learning.

Details

The input data should not contain missing values. Returns a list.

Value

coefficients	a vector of estimated coefficients corresponding to all covariates in z.
loglik	the optimal log partial likelihood.
model.size	the number of covariates that have non-zero coefficient estimates.
select.var	the vector of indices of selected variables, i.e. covariates with non-zero coefficient estimates.
m.stop	the number of updating steps, as pre-specified if cross.validation=FALSE, or giving the optimal risk if cross.validation=TRUE

.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

cv.gboost.default, cv.gboost and gboost.

Examples

```
#  obj <- gboost.default(time=time, delta=delta, z=z, cross.validation=T,
#              nfold=5, track=3, rate=0.01, tol=1.0e-4)
```

plot.cv.gboost *Plotting an object of class 'cv.gboost'.*

Description

For an object of class 'cv.gboost' from function `cv.gboost`, plot cross validation risks versus number of updating steps.

Usage

```
# S3 method for class  'cv.gboost' object
plot(obj, ...)
```

Arguments

`obj`	an object of class 'cv.gboost'.
`...`	other arguments of `plot()`.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

`plot`.

print.cv.gboost *Print a 'cv.gboost' object*

Description

Print a 'cv.gboost' object

Usage

```
## S3 method for class 'cv.gboost'
print(x, ...)
```

Arguments

`x`	fitted `cv.gboost` object.
`...`	additional print arguments.

Value

Returns the call and formula of `cv.gboost` object.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

`cv.gboost`, `print`.

`print.gboost` *Print a 'gboost' object.*

Description

Print a 'gboost' object.

Usage

```
print(x, ...)
```

Arguments

`x`	a fitted object of class 'gboost'.
`...`	additional arguments for `print`.

Value

Returns the call and formula of a 'gboost' object.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

`print`.

`print.summary.cv.gboost` *Print the summary of a 'cv.gboost' object.*

Description

Print the summary of a 'cv.gboost' object with a standard function `summary()`.

Usage

```
# S3 method for summarizing a 'cv.gboost' object
summary(x, ...)
```

Arguments

`x`	fitted 'cv.gboost' object
`...`	additional arguments for `summary()`.

Value

Returns the formula, log partial likelihood, model size, number of updating steps, selected variables and corresponding coefficient estimates.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

`cv.gboost`, `summary`.

Examples

```
# summary(obj2)
```

`print.summary.gboost` *Summarize a 'gboost' object.*

Description

Print the summary of a 'gboost' object with a standard function `summary()`.

Usage

```
# S3 method for printing the summary of a 'gboost' object
summary(x, ...)
```

Arguments

`x`	fitted 'gboost' object.
`...`	addtional arguments for `summary()`.

Value

Returns the formula, log partial likelihood, model size, number of updating steps, selected variables and corresponding coefficient estimates.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

`gboost`, `summary`.

Examples

```
# summary(obj1)
```

summary.cv.gboost *Summarize a 'cv.gboost' object.*

Description

Summarize a 'cv.gboost' object.

Usage

```
summary.cv.gboost(x, ...)
```

Arguments

x	fitted 'cv.gboost' object.
...	additional arguments

Value

Returns an object of class 'summary.cv.gboost'.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

cv.gboost

summary.gboost *Summarize a 'gboost' object.*

Description

Summarize a 'gboost' object.

Usage

```
summary.gboost(x, ...)
```

Arguments

x	fitted 'gboost' object.
...	additional arguments

Value

Returns an object of class 'summary.gboost'.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

`gboost`

Survl *Combining observed survival time and censoring indicators*

Description

Combining observed survival time and censoring indicators for further use in `cv.gboost` and `gboost`.

Usage

```
Survl(time, delta)
```

Arguments

`time`	a vector of observed survival time
`delta`	a vector of censoring indicators of the same length as `time`.

Value

Returns a matrix with two columns, `time` and `delta`.

Author(s)

Lu Xia, Zhi (Kevin) He

Maintainer: Lu Xia <luxia@umich.edu>

References

He, K., Li, Y., Zhu, J., Liu, H., Lee, E.J., Amos, I.C., Wei, Q. and Li, Y. (2015) *High-dimensional variable selection with error control for genome-wide association studies*.

See Also

`cv.gboost`

Examples

```
# Survl(time,delta)
```

Index