Package ‘SurvBoost’

September 20, 2019

Type Package
Title Gradient Boosting for Survival Data
Version 0.1.2
Author Emily Morris, Kevin He, Yanming Li, Yi Li, Jian Kang
Maintainer Emily Morris <emorrisl@umich.edu>
Description Gradient boosting for optimizing loss functions of different types of survival data, including stratified data.
Emily Morris, Kevin He, Yanming Li, Yi Li, Jian Kang (2018) <arXiv:1803.07715> see for description of methods.
License GPL-3
LazyData TRUE
LinkingTo Rcpp, RcppArmadillo, RcppParallel
RoxygenNote 6.1.1
Imports Rcpp, mvtnorm, ggplot2, plyr, reshape2, directlabels
NeedsCompilation yes
Depends R (>= 2.10), survival
Repository CRAN
Date/Publication 2019-09-20 13:10:06 UTC

R topics documented:

boosting_core
inference.boosting
plot.boosting
predict.boosting
print.boosting
simulate_survival_cox
strata.boosting
summary.boosting

Index
boosting_core  

Boosting core function

Description

This function allows you to use gradient boosting for variable selection.

Usage

boosting_core(formula, data, rate, num_iter = 500,  
control_method = NULL, control_parameter = NULL,  
censoring_type = "right")

Arguments

formula  
a formula object with a response value using the Surv function.

data  
a data.frame containing all variables specified in the formula.

rate  
the desired update rate used in the boosting algorithm.

num_iter  
an integer used as the number of iterations of the boosting algorithm. Default value is 500.

control_method  
specifies stopping method, options include: cv, num_selected, likelihood, BIC, AIC. Default is NULL, which will use a fixed number of iterations as specified by num_iter.

control_parameter  
is a list with the parameter(s) needed for each corresponding control_method option, the options are "cv_folds", "early_stop", "EBIC_gamma", "num_select", and "likelihood_tol." For cv method "cv_folds" specifies the number of cross validation folds (default is 10). For EBIC and AIC methods, "early_stop" is a TRUE/FALSE value for early stopping (default is FALSE). An additional parameter for the EBIC method is "EBIC_gamma" that is used to specify the penalty term, should be a value between 0 and 1. If using num_selected method, "num_select" will be the desired number of variables to select, should be an integer. If using likelihood as the method, "likelihood_tol" will be the small change in likelihood in which to stop once reached (default is 0.001).

censoring_type  
currently only right censoring is implemented.

Value

a list containing the vector of coefficients ("beta"), variable selection matrix that contains the coefficients at each iteration ("selection_df"), the number of boosting iterations ("mstop"), and other stopping criteria if applicable to selected method. If using method BIC or AIC, the information criteria for each iteration is returned as a vector ("Information Criteria"). If using cross validation for stopping the criteria used for stopping is returned as a numeric vector ("cvrisk").
**inference.boosting**

**Boosting inference function**

**Description**

This function provides post selection inference.

**Usage**

```r
inference.boosting(x)
```

**Arguments**

- `x`: output from boosting_core function.

**Examples**

```r
data <- simulate_survival_cox(true_beta=c(1,1,1,1,0,0,0,0,0,0))
formula <- as.formula("Surv(time,delta) ~ strata(strata_idx) + V1 + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10")
boosting_core(formula, data, rate=0.1, num_iter=500)
boosting_core(formula, data, rate=0.1, control_method="num_selected", control_parameter=list(num_select = 5))
inference.boosting(boosting.output)
```

---

**plot.boosting**

**Boosting plot function**

**Description**

This function allows you to visualize the coefficient paths of the boosting algorithm.

**Usage**

```r
## S3 method for class 'boosting'
plot(x, y = NULL, type = "frequency", ...)
```
predict.boosting

Arguments

x
output from the boosting_core function.
y
y coordinates of plot, default is NULL.
type
specifies type of coefficient plot. Default value is frequency which plots the proportion of variables selected. Alternatively type set to "coefficients" plots the coefficient path for each variable.
...
ignored

Examples

data <- simulate_survival_cox(true_beta=c(1,1,1,1,0,0,0,0,0))
formula <- as.formula("Surv(time,delta) ~ strata(strata_idx) + V1 + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10")
boosting.output <- boosting_core(formula, data, rate=0.1, num_iter=500)
plot.boosting(boosting.output)
plot.boosting(boosting.output, type="coefficients")

predict.boosting

Boosting predict function

Description

This function predicts the hazard ratio for each subject in the input dataset.

Usage

## S3 method for class 'boosting'
predict(object, newdata = NULL, ...)

Arguments

object
output from boosting_core function.
newdata
data.frame used for prediction. Default is NULL and will use data specified for boosting algorithm.
...
ignored

Value

vector of the hazard ratio for each observation relative to the sample average.

Examples

data <- simulate_survival_cox(true_beta=c(1,1,1,1,0,0,0,0,0))
formula <- as.formula("Surv(time,delta) ~ strata(strata_idx) + V1 + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10")
boosting.output <- boosting_core(formula, data, rate=0.1, num_iter=500)
predict.boosting(boosting.output)
print.boosting

Prints the call and coefficients from boosting model selection

Description
This function displays the coefficient estimates of all variables from a model generated with the boosting_core function.

Usage
## S3 method for class 'boosting'
print(x, ...)

Arguments
x output from boosting_core function.
... ignored

Value
list containing the coefficient vector and function call.

Examples
data <- simulate_survival_cox(true_beta=c(1,1,1,1,0,0,0,0,0,0))
formula <- as.formula("Surv(time,delta) ~ strata(strata_idx) + V1 + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10")
boosting.output <- boosting_core(formula, data, rate=0.1, num_iter=500)
print(boosting.output)

collapse

simulate_survival_cox Simulating survival data

Description
This function allows you to simulate stratified survival data.

Usage
simulate_survival_cox(true_beta, base_hazard = "auto", base_hazard_scale = NULL, base_hazard_shape = NULL, num_strata = 10, input_strata_size = 50, z_matrix = NULL, cov_structure = "diag", block_size = 1, rho = NULL, censor_dist = "unif", censor_const = 5, tau = Inf, normalized = F)
simulate_survival_cox

Arguments

true_beta  Contains true parameter values to simulate from.
base_hazard  Baseline hazard distribution. Default is set to exponential. "weibull" is also available.
base_hazard_scale  Scale parameter used if baseline hazard distribution is weibull.
base_hazard_shape  Shape parameter used if baseline hazard distribution is weibull.
num_strata  Number of strata to simulate, default is 10.
input_strata_size  Average sample size of each stratum, default is 50.
z_matrix  Covariate matrix. Default is NULL, will be simulated as multivariate normal if not provided.
cov_structure  Covariance structure. Default is "diag" could also be "ar" for AR1 or "cs" for compound symmetry.
block_size  Block size used for covariance structure. Default value is 1.
rho  Correlation parameter used for "ar" or "cs" covariance structure.
censor_dist  Censoring distribution, default is "unif" for uniform distribution. Exponential distribution is used if set to "exp"
censor_const  Parameter used to specify the censoring distribution. Default value is 5.
tau  Positive scalar used to represent possible follow up time. Default is Inf.
normalized  Logical parameter representing whether or not the covariate matrix should be normalized. Default is FALSE.

Value

a matrix with survival time (time), event indicator (delta), stratification variable (strata_idx), a vector for each variable specified by the true_beta.

Examples

toyData <- simulate_survival_cox(true_beta=c(1,1,1,1,1,0,0,0,0,0),
base_hazard="weibull", base_hazard_scale=rep(1,5), base_hazard_shape=rep(2,5),
um_strata=5, input_strata_size=100, cov_structure="diag", block_size=2,
rho=0.3, censor_dist="unif", censor_const=5, tau=Inf, normalized=FALSE)
any(duplicated(toyData$time))
z <- as.matrix(toyData[-c(1,2,3)])
strata.boosting

Stratification function

Description

This function assists in evaluating whether the supplied variable is useful for stratification when fitting a cox proportional hazards model.

Usage

strata.boosting(x, survival.time, split = "median")

Arguments

x variable that may be used for stratification, can be categorical or continuous.
survival.time vector of survival time corresponding to input vector x.
split specifies how to split a continuous variable. Default is median value.

Value

Generates a plot and table. Table displays the quartiles of the groups of x. A boxplot is also generated to display the distributions of the groups in x visually.

Examples

data <- simulate_survival_cox(true_beta=c(1,1,1,1,1,0,0,0,0,0))
strata.boosting(data$strata_idx, data$time)

summary.boosting

Summary of boosting model selection

Description

This function displays the variables selected from a model generated with the boosting_core function.

Usage

## S3 method for class 'boosting'
summary(object, all_beta = NULL, ...)


Arguments

object  output from boosting_core function.
all_beta default value is FALSE. If this is set to TRUE the coefficient estimates for all the parameters will be printed.
... ignored

Value

list containing the coefficient vector, number of boosting iterations, and resulting formula from the variable selection.

Examples

data <- simulate_survival_cox(true_beta=c(1,1,1,1,0,0,0,0,0,0))
formula <- as.formula("Surv(time,delta) ~ strata(strata_idx) + V1 + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9 + V10")
boosting.output <- boosting_core(formula, data, rate=0.1, num_iter=500)
summary.boosting(boosting.output)
Index

*Topic boosting
  - boosting_core, 2
  - inference.boosting, 3
  - plot.boosting, 3
  - predict.boosting, 4
  - print.boosting, 5
  - simulate_survival_cox, 5
  - strata.boosting, 7
  - summary.boosting, 7

*Topic gradient
  - boosting_core, 2
  - inference.boosting, 3
  - plot.boosting, 3
  - predict.boosting, 4
  - print.boosting, 5
  - simulate_survival_cox, 5
  - strata.boosting, 7
  - summary.boosting, 7

boosting_core, 2

ingression.boosting, 3

plot.boosting, 3
predict.boosting, 4
print.boosting, 5
simulate_survival_cox, 5
strata.boosting, 7
summary.boosting, 7