Package 'rocTree'

March 24, 2019

Title Receiver Operating Characteristic (ROC)-Guided Classification and Survival Tree

Version 1.0.0

Description Receiver Operating Characteristic (ROC)-guided survival trees and forests algorithms are implemented, providing a unified framework for tree-structured analysis with censored survival outcomes. A time-invariant partition scheme on the survivor population was considered to incorporate time-dependent covariates. Motivated by ideas of randomized tests, generalized time-dependent ROC curves were used to evaluate the performance of survival trees and establish the optimality of the target hazard function. The optimality of the target hazard function motivates us to use a weighted average of the time-dependent area under the curve (AUC) on a set of time points to evaluate the prediction performance of survival trees and to guide splitting and pruning. A detailed description of the implemented methods can be found in Sun et al. (2019) <arXiv:1809.05627>.

Depends R (>= 3.4.0)

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports DiagrammeR (>= 1.0.0), parallel, data.tree (>= 0.7.5), graphics, stats, survival (>= 2.38), methods, tibble, dplyr, ggplot2, MASS, flexsurv

URL http://github.com/stc04003/rocTree

BugReports http://github.com/stc04003/rocTree/issues

NeedsCompilation yes

Author Yifei Sun [aut], Mei-Cheng Wang [aut], Sy Han Chiou [aut, cre]

Maintainer Sy Han Chiou <schiou@utdallas.edu>

Repository CRAN

Date/Publication 2019-03-24 19:50:02 UTC

R topics documented:

`rocTree-package` *rocTree:Receiver Operating Characteristic (ROC)-Guided Classification Survival Tree and Forest.*

Description

The `rocTree` package uses a Receiver Operating Characteristic guided classification algorithm to grow and prune survival trees. The `rocTree` package also provides implementation to grow random forest.

Introduction

The `rocTree` package provides implementations to a unified framework for tree-structured analysis with censored survival outcomes. Different from many existing tree building algorithms, the `rocTree` package incorporates time-dependent covariates by constructing a time-invariant partition scheme on the survivor population. The partition-based risk prediction function is constructed using an algorithm guided by the Receiver Operating Characteristic (ROC) curve. Specifically, the generalized time-dependent ROC curves for survival trees show that the target hazard function yields the highest ROC curve. The optimality of the target hazard function motivates us to use a weighted average of the time-dependent area under the curve on a set of time points to evaluate the prediction performance of survival trees and to guide splitting and pruning. Moreover, the `rocTree` package also offers a novel risk prediction forest algorithm, where the ensemble is on unbiased martingale estimating equations.

Methods

The package contains functions to construct ROC-guided survival trees (`rocTree`) and random forest (`rocForest`).

Author(s)

Maintainer: Sy Han Chiou <schiou@utdallas.edu>

Authors:

- Yifei Sun <ys3072@cumc.columbia.edu>
- Mei-Cheng Wang <mcwang@jhu.edu>

See Also

rocTree, rocForest

---

plot.rocForest *Plotting an* rocForest *object*

Description

Plots a tree from an rocForest object. See plot.rocTree for more details.

Usage

```
## S3 method for class 'rocForest'
plot(x, tree = 1L, output = c("graph",
  "visNetwork"), digits = 4, rankdir = c("TB", "BT", "LR", "RL"),
  shape = "ellipse", nodeOnly = FALSE, savePlot = FALSE,
  file_name = "pic.pdf", file_type = "pdf", ...)
```

Arguments

x	an rocForest object.
tree	an integer specifying the n^{th} tree in the forest to print.
output	a string specifying the output type; graph (the default) renders the graph using the grViz function, and visNetwork renders the graph using the visnetwork function.
digits	the number of digits to print.
rankdir	is a character string specifying the direction of the tree flow. The available options are top-to-bottom ("TB"), bottom-to-top ("BT"), left-to-right ("LR"), and right-to-left ("RL"); the default value is "TB".
shape	is a character string specifying the shape style. Some of the available options are "ellipse", "oval", "rectangle", "square", "egg", "plaintext", "diamond", and "triangle". The default value is "ellipse".
nodeOnly	is a logical value indicating whether to display only the node number; the default value is "TRUE".
savePlot	is a logical value indicating whether the plot will be saved (exported); the default value is "FALSE".

file_name	is a character string specifying the name of the plot when "savePlot = TRUE". The file name should include its extension. The default value is "pic.pdf"
file_type	is a character string specifying the type of file to be exported. Options for graph files are: "png", "pdf", "svg", and "ps". The default value is "pdf".
...	arguments to be passed to or from other methods.

---

plot.rocTree *Plotting an* rocTree *object*

Description

Plots an rocTree object. The function returns a dgr_graph object and is rendered in the RStudio Viewer.

Usage

```
## S3 method for class 'rocTree'
plot(x, output = c("graph", "visNetwork"),
  digits = 4, rankdir = c("TB", "BT", "LR", "RL"), shape = "ellipse",
  nodeOnly = FALSE, savePlot = FALSE, file_name = "pic.pdf",
  file_type = "pdf", ...)
```

Arguments

x	an object of class "rocTree", usually returned by the rocTree function.
output	a string specifying the output type; graph (the default) renders the graph using the grViz function, and visNetwork renders the graph using the visnetwork function.
digits	the number of digits to print.
rankdir	is a character string specifying the direction of the tree flow. The available options are top-to-bottom ("TB"), bottom-to-top ("BT"), left-to-right ("LR"), and right-to-left ("RL"); the default value is "TB".
shape	is a character string specifying the shape style. Some of the available options are "ellipse", "oval", "rectangle", "square", "egg", "plaintext", "diamond", and "triangle". The default value is "ellipse".
nodeOnly	is a logical value indicating whether to display only the node number; the default value is "TRUE".
savePlot	is a logical value indicating whether the plot will be saved (exported); the default value is "FALSE".
file_name	is a character string specifying the name of the plot when "savePlot = TRUE". The file name should include its extension. The default value is "pic.pdf"
file_type	is a character string specifying the type of file to be exported. Options for graph files are: "png", "pdf", "svg", and "ps". The default value is "pdf".
...	arguments to be passed to or from other methods.

See Also

See rocTree for creating rocTree objects.

Examples

```
set.seed(1)
dat <- simu(100, 0, 1.3)
library(survival)
system.time(fit <- rocTree(Surv(Time, death) ~ z1 + z2, id = id,
data = dat, control = list(prune = TRUE, nflds = 10)))
plot(fit)
plot(fit, rankdir = "LR", nodeOnly = TRUE)
plot(fit, output = "visNetwork", nodeOnly = TRUE, shape = "box")
```

---

plotTreeHaz *Plotting the estimated hazard function from an rocTree object*

Description

Plot the estimated hazard function from rocTree objects.

Usage

```
plotTreeHaz(x, ghN = NULL)
```

Arguments

x	an object of class "rocTree", usually returned by the rocTree function.
ghN	an optional smoothing parameter used in smoothing hazard functions; the default value is 0.2.

Examples

```
set.seed(1)
dat <- simu(100, 0, 1.3)
library(survival)
system.time(fit <- rocTree(Surv(Time, death) ~ z1 + z2, id = id,
data = dat, control = list(prune = TRUE, nflds = 10)))
plotTreeHaz(fit)
```

predict.rocForest *Predicting based on a* rocForest *model.*

Description

The function gives predicted values with a rocForest fit.

Usage

```
## S3 method for class 'rocForest'
predict(object, newdata, type = c("survival",
  "hazard", "cumHaz"), ...)
```

Arguments

object	is an rocForest object.
newdata	is an optional data frame in which to look for variables with which to predict. If omitted, the fitted predictors are used. If the covariate observation time is not supplied, covariates will be treated as at baseline.
type	is an optional character string specifying whether to predict the survival probability or the hazard rate.
...	for future developments.

See Also

predict.rocTree

predict.rocTree *Predicting based on a* rocTree *model.*

Description

The function gives predicted values with a rocTree fit.

Usage

```
## S3 method for class 'rocTree'
predict(object, newdata, type = c("survival",
  "hazard"), ...)
```

Arguments

object	is an rocTree object.
newdata	is an optional data frame in which to look for variables with which to predict. If omitted, the fitted predictors are used. If the covariate observation time is not supplied, covariates will be treated as at baseline.
type	is an optional character string specifying whether to predict the survival probability or the cumulative hazard rate.
...	for future developments.

Value

Returns a data.frame of the predicted survival probabilities or cumulative hazard.

See Also

predict.rocForest

---

print.rocForest *Printing an* rocForest *object*

Description

Prints a tree from an rocForest object.

Usage

```
## S3 method for class 'rocForest'
print(x, tree = NULL, digits = 5, dt = TRUE, ...)
```

Arguments

x	an rocForest object.
tree	an optional integer specifying the n^{th} tree in the forest to print. The function prints the contents of an rocForest object by default.
digits	the number of digits of numbers to print.
dt	an optional logical vector. If TRUE, tree structure based on data.tree **structure** is printed.
...	for future development.

See Also

rocForest, print.rocTree

print.rocTree *Printing an* rocTree *object*

Description

The function prints an rocTree object. It is a method for the generic function print of class "rocTree".

Usage

```
## S3 method for class 'rocTree'
print(x, digits = 5, dt = TRUE, ...)
```

Arguments

x	an rocTree object.
digits	the number of digits of numbers to print.
dt	an optional logical vector. If TRUE, tree structure based on data.tree **structure** is printed.
...	for future development.

Examples

```
set.seed(1)
dat <- simu(100, 0, 1.3)
library(survival)
system.time(fit <- rocTree(Surv(Time, death) ~ z1 + z2, id = id,
data = dat, control = list(prune = TRUE, nflds = 10)))
fit
print(fit, dt = FALSE)
```

rocForest *ROC-guided Regression Forest*

Description

Fits a "rocForest" model.

Usage

```
rocForest(formula, data, id, subset, splitBy = c("dCON", "CON"),
  control = list())
```

Arguments

formula	a formula object, with the response on the left of a '~' operator, and the terms on the right. The response must be a survival object returned by the 'Surv' function.
data	an optional data frame in which to interpret the variables occurring in the 'formula'.
id	an optional vector used to identify time dependent covariate. If missing, then each individual row of 'data' is presumed to represent a distinct subject and each covariate is treated as a baseline covariate. The length of 'id' should be the same as the number of observations.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
splitBy	a character string specifying the splitting algorithm. The available options are 'CON' and 'dCON' corresponding to the splitting algorithm based on the total concordance measure or the difference in concordance measure, respectively. The default value is 'dCON'.
control	a list of control parameters. See 'details' for important special features of control parameters.

Details

The argument "control" defaults to a list with the following values:

`tau` maximum follow-up time; default value is the 90th percentile of the unique observed survival times.

`M` maximum node number allowed to be in the tree; the default value is 1000.

`hN` smoothing parameter; the default value is "tau / 20".

`minsp` the minimum number of failure required in a node after a split; the default value is 20.

`minsp2` the minimum number of failure required in a terminal node after a split; the default value is 5.

`disc` a logical vector specifying whether the input covariate are discrete (`disc` = 1). The length of "disc" should be the same as the number of covariates.

`parallel` a logical vector specifying whether parallel computing will be applied to grow the random forest; the default value is FALSE.

`parCluster` an integer value specifying the number of CPU cores to be used when `parallel` = TRUE. The default value is half of the number of CPU cores detected.

`B` is the number of survival trees to grow for the random forest; the default value is 500.

`fsz` is a function or a numerical value to specify the size of subsample. The default value is half of the number of subjects, e.g., round(n) / 2, where n is the number of subjects.

Value

An object of S3 class "`rocForest`" representing the fit.

Examples

```
library(survival)
set.seed(1)
dat <- simu(100, 0, 1.3)
fit <- rocForest(Surv(Time, death) ~ z1 + z2, id = id, data = dat,
        control = list(minsp = 3, minsp2 = 1, B = 50))
fit

## Print individual trees
print(fit, 1)
print(fit, 2)
```

`rocTree` *ROC-guided Regression Trees*

Description

Fits a "`rocTree`" model.

Usage

```
rocTree(formula, data, id, subset, splitBy = c("dCON", "CON"),
  control = list())
```

Arguments

`formula`	a formula object, with the response on the left of a '~' operator, and the terms on the right. The response must be a survival object returned by the 'Surv' function.
`data`	an optional data frame in which to interpret the variables occurring in the 'formula'.
`id`	an optional vector used to identify the longitudinal observations of subject's id. The length of 'id' should be the same as the total number of observations. If 'id' is missing, each row of 'data' represents a distinct observation from a subject and all covariates are treated as a baseline covariate.
`subset`	an optional vector specifying a subset of observations to be used in the fitting process.
`splitBy`	a character string specifying the splitting algorithm. The available options are 'CON' and 'dCON' corresponding to the splitting algorithm based on the total concordance measure or the difference in concordance measure, respectively. The default value is 'dCON'.
`control`	a list of control parameters. See 'details' for important special features of control parameters.

Details

The argument "control" defaults to a list with the following values:

`tau` maximum follow-up time; default value is the 90th percentile of the unique observed survival times.

`M` maximum node number allowed to be in the tree; the default value is 1000.

`hN` smoothing parameter; the default value is "tau / 20".

`minsp` the minimum number of failure required in a node after a split; the default value is 20.

`minsp2` the minimum number of failure required in a terminal node after a split; the default value is 5.

`disc` a logical vector specifying whether the input covariate are discrete (`disc` = 1). The length of "disc" should be the same as the number of covariates.

`prune` a logical vector specifying whether to prune the survival tree. If 'TRUE', a cross-validation procedure will be performed to determine the optimal subtree; the default value is FALSE.

`nflds` the number of folds used in the cross-validation. This argument is only needed if `prune` = TRUE. The default value is 10.

`Trace` a logical vector specifying whether to display the splitting path; the default value is FALSE.

`parallel` a logical vector specifying whether parallel computing will be applied in cross-validation when `prune` = TRUE; the default value is FALSE.

`parCluster` an integer value specifying the number of CPU cores to be used when `prune` = TRUE and `parallel` = TRUE. The default value is half of the number of CPU cores detected.

Value

An object of S3 class "`rocTree`" representing the fit, with the following components:

Frame is a data frame describe the resulting tree.

dfFinal estimated hazards at all terminal nodes.

References

Sun Y. and Wang, M.C. (2018+). ROC-guided classification and survival trees. *Technical report*.

See Also

See `print.rocTree` and `plot.rocTree` for printing and plotting an `rocTree`, respectively.

Examples

```
library(survival)
set.seed(1)
dat <- simu(100, 0, 1.3)
fit <- rocTree(Surv(Time, death) ~ z1 + z2, id = id, data = dat,
       control = list(prune = TRUE, nflds = 5))
fit
```

simu *Function to generate simulated data used in the manuscript.*

Description

This function is used to generate simulated data under various settings. Let Z be a p-dimensional vector of possible time-dependent covariates and β be the vector of regression coefficient. The survival times (T) are generated from the hazard function specified as follow:

Scenario 1.1 Proportional hazards model:

$$\lambda(t|Z) = \lambda_0(t)e^{-0.5Z_1+0.5Z_2-0.5Z_3...+0.5Z_{10}},$$

where $\lambda_0(t) = 2t$.

Scenario 1.2 Proportional hazards model with noise variable:

$$\lambda(t|Z) = \lambda_0(t)e^{2Z_1+2Z_2+0Z_3+...+0Z_{10}},$$

where $\lambda_0(t) = 2t$.

Scenario 1.3 Proportional hazards model with nonlinear covariate effects:

$$\lambda(t|Z) = \lambda_0(t)e^{[2\sin(2\pi Z_1)+2|Z_2-0.5|]},$$

where $\lambda_0(t) = 2t$.

Scenario 1.4 Accelerated failure time model:

$$\log(T) = -2 + 2Z_1 + 2Z_2 + \epsilon,$$

where ϵ follows $N(0, 0.5^2)$.

Scenario 1.5 Generalized gamma family:

$$T = e^{\sigma\omega},$$

where $\omega = \log(Q^2 g)/Q$, g follows Gamma(Q^{-2}, 1), $\sigma = 2Z_1, Q = 2Z_2$.

Scenario 2.1 Dichotomous time dependent covariate with at most one change in value:

$$\lambda(t|Z(t)) = \lambda_0(t)e^{2Z_1(t)+2Z_2},$$

where $Z_1(t)$ is the time-dependent covariate: $Z_1(t) = \theta I(t \geq U_0) + (1-\theta)I(t < U_0)$, ,$\theta$ is a Bernoulli variable with equal probability, and U_0 follows a uniform distribution over $[0, 1]$.

Scenario 2.2 Dichotomous time dependent covariate with multiple changes:

$$\lambda(t|Z(t)) = e^{2Z_1(t)+2Z_2},$$

where $Z_1(t) = \theta[I(U_1 \leq t < U_2) + I(U_3 \leq t)] + (1-\theta)[I(t < U_1) + I(U_2 \leq t < U_3)]$, θ is a Bernoulli variable with equal probability, and $U_1 \leq U_2 \leq U_3$ are the first three terms of a stationary Poisson process with rate 10.

Scenario 2.3 Proportional hazard model with a continuous time dependent covariate:

$$\lambda(t|Z(t)) = 0.1e^{Z_1(t)+Z_2},$$

where $Z_1(t) = kt + b$, k and b are independent uniform random variables over $[1, 2]$.

Scenario 2.4 Non-proportional hazards model with a continuous time dependent covariate:

$$\lambda(t|Z(t)) = 0.1 \cdot [1 + \sin\{Z_1(t) + Z_2\}],$$

where $Z_1(t) = kt + b$, k and b follow independent uniform distributions over $[1, 2]$.

Scenario 2.5 Non-proportional hazards model with a nonlinear time dependent covariate:

$$\lambda(t|Z(t)) = 0.1 \cdot [1 + \sin\{Z_1(t) + Z_2\}],$$

where $Z_1(t) = 2kt \cdot \{I(t > 5) - 1\} + b$, k and b follow independent uniform distributions over $[1, 2]$. The censoring times are generated from an independent uniform distribution over $[0, c]$, where c was tuned to yield censoring percentages of 25

Usage

```
simu(n, cen, scenario, summary = FALSE)

trueHaz(dat)

trueSurv(dat)
```

Arguments

`n`	an integer value indicating the number of subjects.
`cen`	is a numeric value indicating the censoring percentage; three levels, 0%, 25%, 50%, are allowed.
`scenario`	can be either a numeric value or a character string. This indicates the simulation scenario noted above.
`summary`	a logical value indicating whether a brief data summary will be printed.
`dat`	is a data.frame prepared by `simu`.

Value

`simu` returns a `data.frame`. The returned data.frame consists of columns:

id is the subject id.

Y is the observed follow-up time.

death is the death indicator; death = 0 if censored.

z1–z10 is the possible time-independent covariate.

k, b, U are the latent variables used to generate $Z_1(t)$ in Scenario 2.1 – 2.5.

The returned data.frame can be supply to `trueHaz` and `trueSurv` to generate the true cumulative hazard function and the survival function, respectively.

Examples

```
set.seed(1)
simu(10, 0.25, 1.2, TRUE)

set.seed(1)
simu(10, 0.50, 2.2, TRUE)
```

Index