Package 'ICcforest'

July 21, 2025

Version 0.5.1

Date 2020-02-16

Title An Ensemble Method for Interval-Censored Survival Data

Author Weichi Yao [aut, cre],
Halina Frydman [aut],
Jeffrey S. Simonoff [aut]

Maintainer Weichi Yao <wy635@stern.nyu.edu>

Depends R (>= 3.4.0), partykit

Imports stats, utils, graphics, survival, icenReg, ipred

Suggests LTRCtrees, inum, parallel

Description Implements the conditional inference forest approach
to modeling interval-censored survival data. It also provides
functions to tune the parameters and evaluate the model fit. See
Yao et al. (2019) <doi:10.48550/arXiv.1901.04599>.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

NeedsCompilation no

Repository CRAN

Date/Publication 2020-02-17 05:30:02 UTC

Contents

ICcforest-package *Construct a conditional inference forest model for interval-censored survival data*

Description

Construct a conditional inference forest model for interval-censored survival data. The main function of this package is ICcforest.

Details

Problem setup and existing methods: In many situations, the survival time cannot be directly observed and it is only known to have occurred in an interval obtained from a sequence of examination times. Methods like the Cox proportional hazards model rely on restrictive assumptions such as proportional hazards and a log-linear relationship between the hazard function and covariates. Furthermore, because these methods are often parametric, nonlinear effects of variables must be modeled by transformations or expanding the design matrix to include specialized basis functions for more complex data structures in real world applications. The function ICtree in the LTRCtrees package provides a conditional inference tree method for interval-censored survival data, as an extension of the conditional inference tree method ctree for right-censored data. Tree estimators are nonparametric and as such often exhibit low bias and high variance. Ensemble methods like bagging and random forest can reduce variance while preserving low bias.

ICcforest model: This package implements ICcforest, which extends the conditional inference forest (see cforest) to interval censored data. ICcforest uses conditional inference survival trees (see ICtree) as base learners. The main function ICcforest fits a conditional inference forest for interval-censored survival data, with parameter mtry tuned by tuneICRF; gettree.ICcforest extracts the i-th individual tree from the established ICcforest objects; and predict.ICcforest computes predictions from ICcforest objects.

See Also

ICcforest, gettree.ICcforest, predict.ICcforest, tuneICRF, sbrier_IC

gettree.ICcforest *Extract an individual tree from an ICcforest object*

Description

Extract the i-th individual tree from the established ICcforest. The resulting object can be printed or plotted, and predictions can be made using it.

Usage

```
## S3 method for class 'ICcforest'
gettree(object, tree = 1L, ...)
```

Arguments

object	an object as returned by ICcforest.
tree	an integer, the number of the tree to extract from the forest.
...	additional arguments.

Value

An object of class party.

Examples

```
#### Example with dataset miceData
library(icenReg)
data(miceData)

## For ICcforest to run, Inf should be set to be a large number, for example, 9999999.
idx_inf <- (miceData$u == Inf)
miceData$u[idx_inf] <- 9999999.

## First, fit an iterval-censored conditional inference forest
Cforest <- ICcforest(formula = Surv(l,u,type="interval2")~grp, data = miceData, ntree = 50L)
## Extract the 50-th tree from the forest
plot(gettree(Cforest, tree = 50L))
```

ICcforest *Fit a conditional inference forest for interval-censored survival data*

Description

An implementation of the random forest and bagging ensemble algorithms utilizing conditional inference trees as base learners for interval-censored survival data.

Usage

```
ICcforest(
  formula,
  data,
  mtry = NULL,
  ntree = 100L,
  applyfun = NULL,
  cores = NULL,
  na.action = na.pass,
  suppress = TRUE,
  trace = TRUE,
  perturb = list(replace = FALSE, fraction = 0.632),
 control = partykit::ctree_control(teststat = "quad", testtype = "Univ", mincriterion =
```

```
  0, saveinfo = FALSE, minsplit = nrow(data) * 0.15, minbucket = nrow(data) * 0.06),
  ...
)
```

Arguments

formula	a formula object, with the response being a Surv object, with form Surv(time1, time2, type="interval2").
data	a data frame containing the variables named in formula.
mtry	number of input variables randomly sampled as candidates at each node for random forest like algorithms. The default mtry is tuned by tuneICRF.
ntree	an integer, the number of the trees to grow for the forest. ntree = 100L is set by default.
applyfun	an optional lapply-style function with arguments function(X, FUN, ...). It is used for computing the variable selection criterion. The default is to use the basic lapply function unless the cores argument is specified (see below). See ctree_control.
cores	numeric. If set to an integer the applyfun is set to mclapply with the desired number of cores. See ctree_control.
na.action	a function which indicates what should happen when the data contain missing values.
suppress	a logical specifying whether the messages from getFitEsts are suppressed. If FALSE, the messages are printed. suppress = TRUE is set by default.
trace	whether to print the progress of the search of the optimal value of mtry when mtry is not specified (see tuneICRF). trace = TRUE is set by default.
perturb	a list with arguments replace and fraction determining which type of resampling, with replace = TRUE referring to the n-out-of-n bootstrap and replace = FALSE referring to sample splitting. fraction is the proportion of observations to draw without replacement.
control	a list of control parameters, see ctree_control. control parameters minsplit, minbucket have been adjusted from the cforest defaults. Other default values correspond to those of the default values used by ctree_control.
...	additional arguments.

Details

ICcforest returns an ICcforest object. The object belongs to the class ICcforest, as a subclass of cforest. This function extends the conditional inference survival forest algorithm in cforest to fit interval-censored survival data.

Value

An object of class ICcforest, as a subclass of cforest.

See Also

predict.ICcforest for prediction, gettree.ICcforest for individual tree extraction, and tuneICRF for mtry tuning.

Examples

```
#### Example with miceData
library(icenReg)
data(miceData)

## For ICcforest to run, Inf should be set to be a large number, for example, 9999999.
miceData$u[miceData$u == Inf] <- 9999999.

## Fit an interval-censored conditional inference forest
Cforest <- ICcforest(Surv(l, u, type = "interval2") ~ grp, data = miceData)
```

---

predict.ICcforest *Predict from an ICcforest model*

Description

Compute predictions from ICcforest objects.

Usage

```
## S3 method for class 'ICcforest'
predict(
  object,
  newdata = NULL,
  OOB = FALSE,
  suppress = TRUE,
  type = c("response", "prob", "weights", "node"),
  FUN = NULL,
  simplify = TRUE,
  scale = TRUE,
  ...
)
```

Arguments

object	an object as returned by ICcforest.
newdata	an optional data frame containing test data.
OOB	a logical specifying whether out-of-bag predictions are desired (only if newdata = NULL).
suppress	a logical specifying whether the messages from getFitEsts are suppressed. If FALSE, the messages are printed. suppress = TRUE is set by default.

type	a character string denoting the type of predicted value returned. For "type = response", the mean of a numeric response, the median survival time for the interval-censored response is returned. For "type = prob", a list with the survival function constructed using the non-parametric maximum likelihood estimator for each observation is returned. "type = weights" returns an integer vector of prediction weights. For type = "node", a list of terminal node ids for each of the trees in the forest is returned.
FUN	a function to compute summary statistics. Predictions for each node must be computed based on arguments (y, w) where y is the response and w are case weights.
simplify	a logical indicating whether the resulting list of predictions should be converted to a suitable vector or matrix (if possible), see cforest.
scale	a logical indicating scaling of the nearest neighbor weights by the sum of weights in the corresponding terminal node of each tree, see cforest.
...	additional arguments.

Value

An object of class ICcforest, as a subclass of cforest.

See Also

sbrier_IC for evaluation of model fit for interval-censored data

Examples

```
library(icenReg)
data(miceData)

## For ICcforest to run, Inf should be set to be a large number, for example, 9999999.
miceData$u[miceData$u == Inf] <- 9999999.

## First, fit an iterval-censored conditional inference forest
Cforest <- ICcforest(formula = Surv(l,u,type="interval2")~grp, data = miceData)
## Predict the survival function constructed using the non-parametric maximum likelihood estimator
Pred <- predict(Cforest, type = "prob")

## Out-of-bag prediction of the median survival time
PredOOB <- predict(Cforest, type = "response", OOB = TRUE)
```

---

sbrier_IC *Model Fit For Interval-Censored Data*

Description

Compute the (integrated) Brier score to evaluate the model fit for interval-censored survival data.

Usage

```
sbrier_IC(
  obj,
  pred,
  btime = range(as.numeric(obj[, 1:2])),
  type = c("IBS", "BS")
)
```

Arguments

obj	an object of class Surv.
pred	predicted values. This can be a matrix of survival probabilities evaluated at a sequence of time points for a set of new data, a list of survfit objects, a list ic_np objects, or a list of ic_sp objects.
btime	a vector of length two indicating the range of times that the scores are computed on. The default btime is set to be the vector of the smallest and the largest values among all left and right endpoints given in obj.
type	a character string denoting the type of scores returned. For "IBS", the integrated Brier score over the btime is returned. For "BS", the Brier score at every left and right endpoint of all censoring intervals that lie within btime is returned.

Value

If type = "IBS", this returns the integrated Brier score.

If type = "BS", this returns the Brier scores.

References

S. Tsouprou. Measures of discrimination and predictive accuracy for interval-censored data. Master thesis, Leiden University. https://www.math.leidenuniv.nl/scripties/MasterTsouprou.pdf.

Examples

```
### Example with dataset miceData
library(survival)
library(icenReg)
data(miceData)

## For proper evaluation, Inf should be set to be a large number, for example, 9999999.
idx_inf <- (miceData$u == Inf)
miceData$u[idx_inf] <- 9999999.

obj <- Surv(miceData$l, miceData$u, type = "interval2")

## Model fit for an NPMLE survival curve with survfit
pred <- survival::survfit(formula = Surv(l, u, type = "interval2") ~ 1, data = miceData)
# Integrated Brier score up to time = 642
sbrier_IC(obj, pred, btime = c(0, 642), type = "IBS")
```

```
## Model fit for a semi-parametric model with icenReg::ic_sp()
pred <- icenReg::ic_sp(formula = Surv(l, u, type = "interval2") ~ 1, data = miceData)
# Integrated Brier score up to the largest endpoints of all censoring intervals in the dataset
sbrier_IC(obj, pred, type = "IBS")

## Model fit for an NPMLE survival curve with icenReg::ic_np()
pred <- icenReg::ic_np(miceData[,c('l', 'u')])
# Brier score computed at every left and right endpoints of all censoring intervals in the dataset
sbrier_IC(obj, pred, type = "BS")
```

`tuneICRF` *Tune mtry to the optimal value with respect to out-of-bag error for an ICcforest model*

Description

Starting with the default value of mtry, search for the optimal value (with respect to Out-of-Bag error estimate) of mtry for ICcforest.

Usage

```
tuneICRF(
  formula,
  data,
  mtryStart = NULL,
  stepFactor = 1.5,
  ntreeTry = 100L,
 control = partykit::ctree_control(teststat = "quad", testtype = "Univ", mincriterion =
  0, saveinfo = FALSE, minsplit = nrow(data) * 0.15, minbucket = nrow(data) * 0.06),
  suppress = TRUE,
  trace = TRUE,
  plot = FALSE,
  doBest = FALSE
)
```

Arguments

`formula`	a formula object, with the response being a `Surv` object, with form `Surv(time1, time2, type="interval2")`.
`data`	a data frame containing the variables named in `Formula`.
`mtryStart`	starting value of `mtry`; default is `sqrt(nvar)`.
`stepFactor`	at each iteration, `mtry` is inflated (or deflated) by this value.
`ntreeTry`	number of trees used at the tuning step.

control	a list with control parameters, see cforest. The default values correspond to those of the default values used by ICcforest.
suppress	a logical specifying whether the messages from getFitEsts are suppressed. If FALSE, the messages are printed. suppress = TRUE is set by default.
trace	whether to print the progress of the search. trace = TRUE is set by default.
plot	whether to plot the out-of-bag error as a function of mtry.
doBest	whether to run an ICcforest using the optimal mtry found.

Value

If doBest=FALSE (default), this returns the optimal mtry value of those searched.

If doBest=TRUE, this returns the ICcforest object produced with the optimal mtry.

See Also

sbrier_IC for evaluation of model fit for interval-censored data when searching for the optimal value of mtry.

Examples

```
### Example with dataset tandmob2
library(icenReg)
data(miceData)

## For ICcforest to run, Inf should be set to be a large number, for example, 9999999.
miceData$u[miceData$u == Inf] <- 9999999.

## Create a new variable to be selected from
miceData$new = rep(1:4)

## Tune mtry
mtryTune <- tuneICRF(Surv(l, u, type = "interval2") ~ grp + new, data = miceData)
```

Index