Package 'recforest'

January 13, 2026

Title Random Survival Forest for Recurrent Events

Version 1.0.2

Description A tool designed to analyze recurrent events when dealing with right-censored data and the potential presence of a terminal event (that prevents further occurrences, like death). It extends the random survival forest algorithm, adapting splitting rules and node estimators to handle complexities of recurrent events. The methodology is fully described in Murris, J., Bouaziz, O., Jakubczak, M., Katsahian, S., & Lavenu, A. (2024) (<https://hal.science/hal-04612431v1/document>).

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.3.2

Suggests frailtypack, knitr, rmarkdown, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Imports cli, dplyr, future, future.apply, magrittr, methods, mets, purrr, reda, stats, survival, tibble, timereg

VignetteBuilder knitr

Depends R (>= 2.10)

LazyData true

NeedsCompilation no

Author Juliette Murris [aut, cre] (ORCID: <https://orcid.org/0000-0002-7017-9865>), Guillaume Desachy [aut] (ORCID: <https://orcid.org/0000-0002-2000-1436>), Colin Fay [aut] (ORCID: <https://orcid.org/0000-0001-7343-1846>), Yohann Mansiaux [aut] (ORCID: <https://orcid.org/0000-0002-8905-6603>), Audrey Lavenu [aut] (ORCID: <https://orcid.org/0000-0002-0049-2397>), Sandrine Katsahian [aut] (ORCID: <https://orcid.org/0000-0002-7261-0671>)

Maintainer Juliette Murris <murris.juliette@gmail.com>

Repository CRAN

Date/Publication 2026-01-13 16:40:02 UTC

Contents

`bladder1_recforest` *bladder1_recforest : Bladder Cancer Recurrences*

Description

Preparation of the survival::bladder1 dataset for the recforest package. Please run `?survival::bladder1` for more information.

Usage

```
bladder1_recforest
```

Format

A data frame with 294 rows (118 individuals) and 8 variables:

id Patient id

t.start Start time

t.stop Stop time

treatment Placebo, pyridoxine (vitamin B6), or thiotepa

number Initial number of tumors (8=8 or more)

size Size (cm) of largest initial tumor

death Death event

event Recurrence event

Source

Script to generate the data can be explored using `browseURL(system.file("generate_bladder1_recforest.R", package = "recforest"))`

is_supported_variable *Check if a Variable is Supported*

Description

This function is a generic method that checks if a given variable is supported. The actual implementation of the check is provided by specific methods for different classes of variables.

Usage

```
is_supported_variable(x)
```

Arguments

x	The variable to be checked.

Value

A logical value indicating whether the variable is supported.

make_decision *Make a Decision Based on Input*

Description

This function serves as a generic method for making decisions based on the input `x` and `value`. It dispatches to the appropriate method depending on the class of `x`.

Usage

```
make_decision(x, value)
```

Arguments

x	An object for which a decision needs to be made.
value	A value that influences the decision-making process.

Value

The result of the decision-making process, which depends on the specific method implementation.

plot.recforest *Plot method for recforest objects*

Description

Plot method for recforest objects

Usage

```
## S3 method for class 'recforest'
plot(x, ...)
```

Arguments

x	An object of class recforest.
...	Additional arguments to be passed to the plot function.

predict.recforest *Predict using a recforest model*

Description

This function generates predictions from a recforest model given a set of input features.

Usage

```
## S3 method for class 'recforest'
predict(
  object,
  newdata,
  id_var,
  covariates,
  time_vars = c("t.start", "t.stop"),
  death_var = NULL,
  ...
)
```

Arguments

object	A recforest model object.
newdata	A data frame containing the input features.
id_var	The name of the column containing the unique identifier for each subject.
covariates	A character vector containing the names of the columns to be used as predictors in the model.

time_vars	A length-2 character vector containing the names of the columns representing the start and stop times (default "t.start" and "t.stop").
death_var	The name of the column containing the death indicator or other any terminal event (optional).
...	Optional parameters to be passed to the low level function

Details

The predict_recforest function utilizes the ensemble of trees in the recforest model to generate predictions for new data. For each observation in newdata, the function aggregates the predictions from all trees in the recforest to provide a robust estimate.

Depending on the method specified during the initial training of the recforest model, the algorithm employs different prediction strategies:

- For standard recurrent event data, the function outputs the Nelson-Aalen estimates of the mean cumulative function.
- In the presence of terminal events, the function outputs the Ghosh-Lin estimates of the mean cumulative function.

The predictions represent the expected mean number of recurrent events for each individual at the end of the follow-up period.

Value

A vector of expected mean cumulative number of recurrent events per individual at the end of follow-up.

References

Cook, R. J., & Lawless, J. F. (1997). Marginal analysis of recurrent events and a terminating event. Statistics in medicine, 16(8), 911-924.

Ghosh, D., & Lin, D. Y. (2002). Marginal regression models for recurrent and terminal events. Statistica Sinica, 663-688.

Ishwaran, H., Kogalur, U. B., Blackstone, E. H., & Lauer, M. S. (2008). Random survival forests.

Examples

```
if (interactive()) {
  data("bladder1_recforest")
  trained_forest <- train_forest(
    data = bladder1_recforest,
    id_var = "id",
    covariates = c("treatment", "number", "size"),
    time_vars = c("t.start", "t.stop"),
    death_var = "death",
    event = "event",
    n_trees = 5,
    n_bootstrap = 70,
    mtry = 2,
```

```
    minsplit = 3,
    nodesize = 15,
    method = "NAa",
    min_score = 5,
    max_nodes = 20,
    seed = 111,
    parallel = FALSE,
    verbose = FALSE
  )
  predictions <- predict(
    trained_forest,
    newdata = bladder1_recforest,
    id_var = "id",
    covariates = c("treatment", "number", "size"),
    time_vars = c("t.start", "t.stop"),
    death_var = "death"
  )
}
```

`print.recforest` *Print method for recforest objects*

Description

Print method for recforest objects

Usage

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

Arguments

`x`	An object of class `recforest`.
`...`	Additional arguments to be passed to the plot print

`summary.recforest` *Summary Method for recforest Objects*

Description

This function provides a summary of a recforest object by printing its metrics.

Usage

```
## S3 method for class 'recforest'
summary(object, ...)
```

Arguments

object	An object of class recforest.
...	Additional arguments to be passed to the summary function.

Value

The function prints the metrics of the recforest object.

---

train_forest *Train a Recforest Model*

Description

This function trains a recforest model using the provided data and parameters.

Usage

```
train_forest(
  data,
  id_var,
  covariates,
  event,
  time_vars = c("t.start", "t.stop"),
  death_var = NULL,
  n_trees,
  n_bootstrap = NULL,
  seed = NULL,
  mtry,
  minsplit,
  nodesize,
  method,
  min_score,
  max_nodes,
  parallel = FALSE,
  verbose = TRUE
)
```

Arguments

data	A data frame containing the dataset to be used for training the model.
id_var	The name of the column containing the unique identifier for each subject.
covariates	A character vector containing the names of the columns to be used as predictors in the model.
event	The name of the column containing the recurrent event indicator.
time_vars	A length-2 character vector containing the names of the columns representing the start and stop times (default "t.start" and "t.stop").

death_var	The name of the column containing the death indicator or other any terminal event (optional).
n_trees	The number of trees to be trained in the recforest model.
n_bootstrap	The number of bootstrap samples to be used for training each tree (in-bag sample). If not provided, it is set to 2/3 of the sample size (in term of number of unique id_var).
seed	An optional seed value to be used for reproducibility purpose (NULL by default).
mtry	The number of candidate variables randomly drawn at each node of the trees. This parameter should be tuned by minimizing the OOB error.
minsplit	The minimal number of events required to split the node. Cannot be smaller than 2.
nodesize	The minimal number of subjects required in both child nodes to split. Cannot be smaller than 1.
method	The method to be used for training the model. Currently, the following methods are supported : either "NAa" for Nelson-Aalen method, with no terminal event and no longitudinal time-dependent features; either "GL" for Ghosh-Lin modelization step with a terminal event and/or at least one longitudinal time-dependent feature.
min_score	The minimum score required to split a node. This parameter is used only when the method is set to "NAa".
max_nodes	The maximum number of nodes per tree.
parallel	A logical value indicating whether to use parallel processing for training the trees.
verbose	A logical value indicating whether to print progress messages.

Details

The recforest model aggregates predictions over an ensemble of trees, each constructed using a set of decision nodes based on specific splitting rules. At each node, a subset of predictors is randomly selected, and an optimal split is determined using an appropriate statistical test. Depending on the specified method, the algorithm employs different statistical tests to find the best split:

- For standard recurrent event data, the pseudo-score test statistic is used to compare two Nelson-Aalen estimates of the mean cumulative function.
- In the presence of terminal events and/or longitudinal variables, the Ghosh-Lin model is utilized to obtain the Wald test statistic, which provides a more accurate assessment of the split. The trees grow until they meet the stopping criteria, which include a minimum number of events (minsplit) and a minimum number of individuals in terminal nodes (nodesize). The final model is an ensemble of these trees, which helps to reduce overfitting and improve predictive performance by averaging the results on the out-of-bag sample.

Value

A list containing the following elements:

trees	A list of trained trees.
tree_metrics	A list of metrics for each tree.
metrics	A summary of the metrics for all trees.
columns	A list of column names used in the training.
params	A list of parameters used to set the model.
n_indiv	Number of individuals in the dataset.
n_predictors	Number of predictors used in the model.
n_trees	Number of trees trained.
n_bootstrap	Number of bootstrap samples used to grow each tree.
time	Computation time used to train the model.

References

Cook, R. J., & Lawless, J. F. (1997). Marginal analysis of recurrent events and a terminating event. Statistics in medicine, 16(8), 911-924.

Ghosh, D., & Lin, D. Y. (2002). Marginal regression models for recurrent and terminal events. Statistica Sinica, 663-688.

Ishwaran, H., Kogalur, U. B., Blackstone, E. H., & Lauer, M. S. (2008). Random survival forests.

Examples

```
if (interactive()) {
  data("bladder1_recforest")
  # To parallel computing
  # n_cores <- min(future::availableCores(), n_trees)
  # future::plan(future::multisession)
  trained_forest <- train_forest(
    data = bladder1_recforest,
    id_var = "id",
    covariates = c("treatment", "number", "size"),
    time_vars = c("t.start", "t.stop"),
    death_var = "death",
    event = "event",
    n_trees = 5,
    n_bootstrap = 70,
    mtry = 2,
    minsplit = 3,
    nodesize = 15,
    method = "NAa",
    min_score = 5,
    max_nodes = 20,
    seed = 111,
    parallel = FALSE,
    verbose = FALSE
  )
  print(trained_forest)
  summary(trained_forest)
}
```

Index