

Package: RLT (via r-universe)

September 30, 2026

Type Package

Title Reinforcement Learning Trees

Version 6.1.1

Date 2026-07-02

Description Random forest with a variety of additional features for regression, classification, and survival analysis. Features include parallel computing with OpenMP, reproducibility with random seeds, variance and confidence band estimations using U-statistics, embedded model for selecting splitting variables and constructing linear combination splits, permutation and distribution-based variable importance, observation and variable weights, and subject tracking across trees.

Author Ruoqing Zhu [aut, cre, cph]
(<https://orcid.org/0000-0002-0753-5716>), Sarah Formentini [aut], Haowen Zhou [ctb], Tianning Xu [ctb], Zhechao Huang [ctb]

Suggests testthat, knitr, rmarkdown, mgcv

VignetteBuilder knitr

License GPL (>= 3)

Encoding UTF-8

Imports Rcpp (>= 1.0.9), stats

LinkingTo Rcpp, RcppArmadillo, dqrng, BH, sitmo

BugReports <https://github.com/teazrq/RLT/issues>

URL <https://github.com/teazrq/RLT>, <https://teazrq.github.io/RLT/>

Roxygen list(markdown = TRUE)

RoxygenNote 8.0.0

Repository <https://teazrq.r-universe.dev>

Date/Publication 2026-07-02 15:51:24 UTC

RemoteUrl <https://github.com/teazrq/rlt>

RemoteRef HEAD

RemoteSha b86125e332979d8dc8ddb1c3203b6b81fd822262

Contents

cindex	2
forest.kernel	3
get.one.tree	4
get.surv.band	5
importance	6
predict.RLT	7
print.importance.RLT	9
print.RLT	9
RLT	10
Index	15

cindex	<i>C-index</i>
--------	----------------

Description

Calculate c-index for survival data

Usage

cindex(y, censor, pred)

Arguments

- y survival time
- censor The censoring indicator if survival model is used
- pred the predicted value for each subject

Value

c-index

Examples

```
set.seed(42)
n <- 100
x <- matrix(rnorm(n * 5), ncol = 5)
y <- rexp(n, rate = exp(rowSums(x[, 1:2])))
censor <- rbinom(n, 1, 0.7)
fit <- RLT(x, y, censor = censor, model = "survival", ntrees = 100)
# Use cumulative hazard at last timepoint as risk score
risk <- fit$Prediction[, ncol(fit$Prediction)]
cindex(y, censor, risk)
```

forest.kernel	<i>random forest kernel</i>
---------------	-----------------------------

Description

Get random forest induced kernel weight matrix of testing samples or between any two sets of data. This is an experimental feature. Use at your own risk.

Usage

```
forest.kernel(
  object,
  X1 = NULL,
  X2 = NULL,
  vs.train = FALSE,
  oob = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

object	A fitted RLT object.
X1	The dataset for prediction. This calculates an $n_1 \times n_1$ kernel matrix of X1.
X2	The dataset for reference/training. If X2 is supplied, then calculate an $n_1 \times n_2$ kernel matrix. If vs.train is used, then this must be the original training data.
vs.train	To calculate the kernel weights with respect to the training data. This is slightly different than supplying the training data to X2 due to re-samplings of the training process. To use this feature, you must specify <code>resample.track = TRUE</code> in <code>param.control</code> when fitting the forest
oob	Logical. If TRUE, compute the OOB (out-of-bag) self-kernel, which counts co-occurrence only from trees where both observations are OOB. This eliminates the self-contamination bias that arises when in-bag observations influence tree structure. Requires <code>resample.track = TRUE</code> and X1 to be the original training data. X2 must be NULL (OOB kernel is defined for self-kernel only). The returned list contains three matrices: Kernel (normalized co-occurrence in $[0, 1]$), N (number of trees where both are OOB), and C (number of trees where both are OOB and share a leaf).
verbose	Whether fitting should be printed.
...	... Additional arguments.

Value

A list containing the kernel matrix. For `oob = TRUE`, the list also contains N (OOB co-occurrence count) and C (OOB leaf-sharing count).

Examples

```
set.seed(42)
x <- matrix(rnorm(200 * 5), ncol = 5)
y <- rowSums(x[, 1:2]) + rnorm(200)
fit <- RLT(x, y, ntrees = 100)
K <- forest.kernel(fit, X1 = x[1:5, ])
print(K$Kernel[1:3, 1:3])
```

get.one.tree

Print a single tree

Description

Print a single fitted tree from a forest object

Usage

```
get.one.tree(x, tree = 1, ...)
```

Arguments

x	A fitted RLT object
tree	the tree number, starting from 1 to ntrees.
...	...

Value

A data.frame with columns: Node (depth, BFS), NodeType (Split=1, Leaf=-1), SplitVar, SplitValue, LeftNode, RightNode, N (sample count). Model-specific columns include YAvg (regression), Prob (classification), or Hazard/SurvProb (survival).

Examples

```
set.seed(42)
x <- matrix(rnorm(300 * 5), ncol = 5)
y <- rowSums(x[, 1:2]) + rnorm(300)
fit <- RLT(x, y, ntrees = 50)
get.one.tree(fit, tree = 1)
```

get.surv.band

get.surv.band

Description

Calculate the survival function (two-sided) confidence band from a RLT survival prediction.

Usage

```
get.surv.band(
  x,
  i = 0,
  alpha = 0.05,
  approach = "naive",
  nsim = 5000,
  k_rank = 10,
  k_mode = c("fixed", "proportion"),
  k_prop = 0.99,
  ...
)
```

Arguments

x	A RLT prediction object. Must be from a forest with var.mode != "none".
i	Observation number in the prediction. Default to calculate all (i = 0).
alpha	alpha level for interval (alpha/2, 1 - alpha/2).
approach	Confidence band approach: - naive: marsd = sqrt(diag(Cov)), MC band using full covariance. - smoothed: GAM-smoothed rank-K covariance + eigenvalue-ratio weighted residual correction.
nsim	Number of simulations for estimating the Monte Carlo critical value.
k_rank	Rank truncation K used for the smooth low-rank covariance AND GAM basis size.
k_mode	Rank selection mode: "fixed" (use k_rank) or "proportion" (auto-select by eigenvalue cumulative ratio).
k_prop	Proportion threshold (0,1] for cumulative eigenvalue ratio when k_mode = "proportion".
...	Further arguments (currently not used).

Value

An object of class c("RLT", "band", "surv") with components: lower (lower bound), upper (upper bound), and timepoints (evaluation grid). If i = 0, a list of such objects for all observations.

Examples

```
set.seed(42)
n <- 100
x <- matrix(rnorm(n * 5), ncol = 5)
y <- rexp(n, rate = exp(rowSums(x[, 1:2])))
censor <- rbinom(n, 1, 0.7)
fit <- RLT(x, y, censor = censor, model = "survival",
           ntrees = 200, var.mode = "matched")
pred <- predict(fit, testx = x[1:3, ], var.est = TRUE)
band <- get.surv.band(pred, i = 1, alpha = 0.05)
```

importance	<i>Variable Importance Summary</i>
------------	------------------------------------

Description

Extract variable importance from a fitted RLT model. When variance estimation was enabled via `var.mode`, standard deviations, Z-scores, and significance codes are also reported. Negative variance estimates yield NA for SD, Z, and significance.

Usage

```
importance(object, ...)
```

Arguments

<code>object</code>	A fitted RLT object from RLT .
<code>...</code>	Additional arguments (unused).

Value

A data.frame with columns:

- Variable: variable name
- VI: variable importance
- SD: standard deviation of VI (NA if not estimated or negative variance)
- Z: Z-score (VI / SD, NA if SD is NA)
- Sig: significance code ("" if not estimated or negative variance)

Significance codes: *** $|Z| \geq 2.58$, ** $|Z| \geq 1.96$, * $|Z| \geq 1.64$.

Examples

```
## Not run:
fit <- RLT(x, y, model = "classification", importance = TRUE, var.mode = TRUE)
importance(fit)

## End(Not run)
```

predict.RLT	<i>prediction using RLT</i>
-------------	-----------------------------

Description

Predict the outcome (regression, classification or survival) using a fitted RLT object

Usage

```
## S3 method for class 'RLT'
predict(
  object,
  testx = NULL,
  var.est = FALSE,
  var.mode = NULL,
  keep.all = FALSE,
  ncores = 1,
  verbose = 0,
  band.grid.size = 0,
  ...
)
```

Arguments

object	A fitted RLT object
testx	The testing samples, must have the same structure as the training samples
var.est	Whether to estimate the variance of each testing data. The original forest must be fitted with <code>var.mode != "none"</code> . For survival forests, calculates the covariance matrix over all observed time points and calculates critical value for the confidence band.
var.mode	Variance estimation mode for prediction. Can be "none", "matched", "IJ", or "jack". If NULL (default), uses the mode from the fitted object. Only used when <code>var.est = TRUE</code> . "matched" requires the forest to be fitted with <code>var.mode = "matched"</code> . "IJ" and "jack" require the forest to have been fitted with re-sample tracking (automatically enabled when using IJ or jack variance).
keep.all	whether to keep the prediction from all trees. Warning: this can occupy a large storage space, especially in survival model
ncores	number of cores
verbose	print additional information
band.grid.size	An integer specifying the number of time points for confidence band calculation. Default is 0, which uses all unique failure time points. If a positive integer is provided, a subset of time points will be selected using quantiles, skipping the earliest 5% of time points to improve stability.
...	...

Value

A RLT prediction object, constructed as a list consisting

Prediction	Prediction
Variance	if var.est = TRUE and the fitted object is var.mode != "none"

For Survival Forests

hazard	predicted hazard functions
CumHazard	predicted cumulative hazard function
Survival	predicted survival function
Allhazard	if keep.all = TRUE, the predicted hazard function for each observation and each tree
AllCHF	if keep.all = TRUE, the predicted cumulative hazard function for each observation and each tree
Cov	if var.est = TRUE and the fitted object is var.mode != "none". For each test subject, a matrix of size NFail×NFail where NFail is the number of observed failure times in the training data
Var	if var.est = TRUE and the fitted object is var.mode != "none". Marginal variance for each subject
timepoints	ordered observed failure times from the training data
MarginalVar	if var.est = TRUE and the fitted object is var.mode != "none". Marginal variance for each subject from the Cov matrix projected to the nearest positive definite matrix
MarginalVarSmooth	if var.est = TRUE and the fitted object is var.mode != "none". Marginal variance for each subject from the Cov matrix projected to the nearest positive definite matrix and then smoothed using Gaussian kernel smoothing
CVproj	if var.est = TRUE and the fitted object is var.mode != "none". Critical values to calculate confidence bands around cumulative hazard predictions at several confidence levels. Calculated using MarginalVar
CVprojSmooth	if var.est = TRUE and the fitted object is var.mode != "none". Critical values to calculate confidence bands around cumulative hazard predictions at several confidence levels. Calculated using MarginalVarSmooth

Examples

```
set.seed(42)
x <- matrix(rnorm(300 * 5), ncol = 5)
y <- rowSums(x[, 1:2]) + rnorm(300)
fit <- RLT(x, y, ntrees = 100)
pred <- predict(fit, testx = x[1:5, ])
print(pred$Prediction)
```

```
print.importance.RLT
```

Print Importance Summary

Description

Print method for importance.RLT objects.

Usage

```
## S3 method for class 'importance.RLT'
print(x, digits = 4, ...)
```

Arguments

x	An importance.RLT object.
digits	Number of digits for formatting. Default: 4.
...	Additional arguments (unused).

Examples

```
set.seed(42)
x <- matrix(rnorm(100 * 5), ncol = 5)
y <- rowSums(x[, 1:2]) + rnorm(100)
fit <- RLT(x, y, ntrees = 50, importance = TRUE)
print(importance(fit))
```

```
print.RLT
```

Print a RLT object

Description

Print a RLT object

Usage

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

Arguments

x	A fitted RLT object
...	...

Examples

```
set.seed(42)
x <- matrix(rnorm(100 * 5), ncol = 5)
y <- rowSums(x[, 1:2]) + rnorm(100)
fit <- RLT(x, y, ntrees = 50)
print(fit)
```

RLT

*Reinforcement Learning Trees***Description**

Fit models for regression, classification and survival analysis using reinforced splitting rules. The model fits regular random forest models by default unless the parameter `reinforcement` is set to "TRUE". Using `reinforcement = TRUE` activates embedded model for splitting variable selection and allows linear combination split. To specify parameters of embedded models, see definition of `param.control` for details.

Usage

```
RLT(
  x,
  y,
  censor = NULL,
  model = NULL,
  ntrees = if (reinforcement) 100 else 500,
  mtry = max(1, as.integer(ncol(x)/2)),
  nmin = 5,
  alpha = 0,
  nsplit = 0,
  resample.replace = TRUE,
  resample.prob = if (resample.replace) 1 else 0.8,
  resample.preset = NULL,
  obs.w = NULL,
  var.prob = NULL,
  importance = FALSE,
  reinforcement = FALSE,
  linear.comb = 1,
  linear.comb.method = "default",
  split.rule = "default",
  var.mode = "none",
  param.control = list(),
  ncores = 0,
  verbose = 0,
  seed = NULL,
  ...
)
```

Arguments

<code>x</code>	A matrix or data.frame of features. If <code>x</code> is a data.frame, then all factors are treated as categorical variables, which will go through an exhaustive search of splitting criteria.
<code>y</code>	Response variable. a numeric/factor vector.
<code>censor</code>	Censoring indicator if survival model is used.
<code>model</code>	The model type: "regression", "classification", or "survival". Quantile forest is not yet implemented.
<code>ntrees</code>	Number of trees, <code>ntrees = 100</code> if reinforcement is used and <code>ntrees = 500</code> otherwise.
<code>mtry</code>	Number of randomly selected variables used at each internal node. Default: <code>max(1, floor(p/2))</code> .
<code>nmin</code>	Terminal node size. Splitting will stop when the internal node size is less equal to <code>nmin</code> . Default: 5.
<code>alpha</code>	Minimum proportion of samples (of the parent node) enforced in each child node. Default: 0 (no constraint). Clamped to the range 0 to 0.5.
<code>nsplit</code>	Number of random cutting points to compare for each variable at an internal node. Default: 0 (use all unique values, i.e., best split). When <code>nsplit > 0</code> , random cutting points are generated.
<code>resample.replace</code>	Whether the in-bag samples are obtained with replacement.
<code>resample.prob</code>	Proportion of in-bag samples.
<code>resample.preset</code>	A pre-specified matrix for in-bag data indicator/count matrix. It must be an $n \times \text{ntrees}$ matrix with integer entries. Positive number indicates the number of copies of that observation (row) in the corresponding tree (column); zero indicates out-of-bag; negative values indicates not being used in either. Extremely large counts should be avoided. The sum of each column should not exceed n .
<code>obs.w</code>	Observation weights. The weights will be used for calculating the splitting scores, such as a weighted variance reduction or weighted gini index. But they will not be used for sampling observations. In that case, one can pre-specify <code>resample.preset</code> instead for balanced sampling, etc. For survival analysis, observation weights are supported in the "logrank", "suplogrank", and "coxgrad" splitting rules. Weighted logrank and suplogrank use a variance estimator that accounts for the observation weights.
<code>var.prob</code>	Variable probabilities for split variable selection. A numeric vector of length p (number of predictors) with non-negative weights. When supplied, <code>mtry</code> variables are sampled <i>without replacement</i> with probabilities proportional to these weights at each internal node. This effectively up-weights or down-weights individual predictors during tree construction. Works for all models (regression, classification, survival). The vector does not need to sum to 1; it is internally normalized. If NULL (default), uniform sampling is used.
<code>importance</code>	Whether to calculate variable importance measures. When set to "TRUE" (or "permute"), the calculation follows Breiman's original permutation strategy. If

	set to "distribute", then it sends the oob data to both child nodes with weights proportional to their sample sizes. Hence the final prediction is a weighted average of all possible terminal nodes that a perturbed observation could fall into. This feature is currently only available in regression and classification models.
reinforcement	Should reinforcement splitting rule be used. Default is "FALSE", i.e., regular random forests with marginal search of splitting variable. When it is activated, an embedded model is fitted to find the best splitting variable or a linear combination of them, if linear.comb ≥ 1 . They can also be specified in param.control.
linear.comb	Number of variables to combine in each linear combination split. Default is 1 (standard axis-aligned splits). See also linear.comb.method and param.control.
linear.comb.method	Method for constructing linear combinations: "default", "coxph" (Cox PH loading, survival only), or "naive" (covariance-based loading). See param.control.
split.rule	Splitting criterion. Default "default" selects the standard rule for each model. For survival: "logrank", "suplogrank", or "coxgrad". See param.control.
var.mode	Variance estimation mode. Default is "none" (no variance estimation). Set to "matched" or TRUE to use matched-sample U-statistic decomposition for prediction variance and variable importance variance. When active, several resampling parameters are automatically adjusted. Equivalent to setting param.control = list(var.mode = "matched"). See param.control for full details.
param.control	A list of additional parameters. This can be used to specify other features in a random forest or set embedded model parameters for reinforcement splitting rules. Using reinforcement = TRUE will automatically generate some default tuning for the embedded model. Reinforcement is available for regression, classification, and survival models. They are not necessarily optimized. • embed.ntrees: number of trees in the embedded model. Default: 50. • embed.mtry: proportion of variables for embedded splits. Default: 0.5. • embed.nmin: terminal node size for embedded model. Default: 5. • embed.nsplits: number of random cutting points. Default: 3. • embed.resample.replace: whether to sample with replacement. Default: TRUE. • embed.resample.prob: proportion of samples (of the internal node) in the embedded model. Default: 0.9. • embed.mute: variables to mute per split. If ≥ 1: exact count; if < 1: proportion. Default: 0 (no muting). • embed.protect: number of top variables to protect from muting. Default: ceiling(log(n)). • embed.threshold: threshold, as a fraction of the best VI, for being included in the protected set at an internal node. Default: 0.25. • linear.comb: number of variables to use in linear combination splits. Requires reinforcement = TRUE. Default: 1 (no linear combination). • linear.comb.method: method for constructing linear combination splits. Regression: "naive" (1), "lm" (2), "pca" (3), "sir" (4, default). Classification: "lda" (1, default), "naive" (2), "random" (3), "logistic" (4).

- `time.grid.size`: number of unique time points for survival estimation (default 0 = all). See `time.grid.size` argument for details.

See `\code{linear.comb}` and `\code{linear.comb.method}` under `\code{param.control}` documentation above.

`\code{split.rule}` specifies the splitting criterion for each model type.

```
\itemize{
  \item \strong{Regression}: \code{"var"} (variance reduction, default and only)
  \item \strong{Classification}: \code{"gini"} (Gini index, default and only)
  \item \strong{Survival}: \code{"logrank"} (default), \code{"suplogrank"}
}
```

Internally mapped to integers: `var=1`, `gini=1`, `logrank=1`, `suplogrank=2`, `cox=3`.

`\code{resample.track}` indicates whether to keep track of which observations are used in each tree. This is required for variance estimation (via `\code{var.mode}`).

`\code{var.mode}` specifies the variance estimation method to prepare during model fitting. Currently available methods:

```
\itemize{
  \item \code{"none"} (default): No variance estimation.
  \item \code{"matched"}: Uses matched-sample U-statistic decomposition (Xu, Zhu & Shao, 2023) for prediction variance and variable importance variance. Also used for confidence band in survival models (Formentini, Liang & Zhu, 2023).
}
```

Specifying `\code{var.mode = TRUE}` is equivalent to `\code{var.mode = "matched"}`.

When `\code{var.mode}` is not `\code{"none"}`, the following parameters are automatically adjusted if not already set:

```
\itemize{
  \item \code{resample.preset} is constructed automatically
  \item \code{resample.replace} is set to \code{FALSE}
  \item \code{resample.prob} is set to 0.5
  \item \code{resample.track} is set to \code{TRUE}
  \item \code{importance} is set to \code{"distribute"}
}
```

It is recommended to use a very large `\code{ntrees}`, e.g., 10000 or larger. For `\code{resample.prob}` greater than 0.5, one should consider the bootstrap approach in Xu, Zhu & Shao (2023).

`\code{time.grid.size}` specifies the number of unique time points used for survival estimation. By default (0), all observed failure times are used. Setting a smaller number (e.g., 50) can speed up computation for large datasets. The time points are selected at

evenly spaced quantiles of the observed failure times, always including the minimum and maximum failure times.

ncores	Number of CPU logical cores. Default is 0 (using all available cores).
verbose	Whether info should be printed.
seed	Random seed number to replicate a previously fitted forest. Internally, the xoshiro256++ generator is used. If not specified, a seed will be generated automatically and recorded.
...	Additional arguments.

Value

A RLT fitted object, constructed as a list consisting

FittedForest	Fitted tree structures
VarImp	Variable importance measures, if importance = TRUE
Prediction	Out-of-bag prediction
Error	Out-of-bag prediction error, adaptive to the model type
ObsTrack	Provided if <code>resample.track = TRUE</code> , <code>var.mode != "none"</code> , or if <code>resample.preset</code> was supplied. This is an $n \times \text{ntrees}$ matrix that has the same meaning as <code>resample.preset</code> .

For classification forests, these items are further provided or will replace the regression version

NClass	The number of classes
Prob	Out-of-bag predicted probability

For survival forests, these items are further provided or will replace the regression version

timepoints	ordered observed failure times
NFail	The number of observed failure times
Prediction	Out-of-bag prediction of hazard function

References

- Zhu, R., Zeng, D., & Kosorok, M. R. (2015) "Reinforcement Learning Trees." Journal of the American Statistical Association. 110(512), 1770-1784.
- Xu, T., Zhu, R., & Shao, X. (2023) "On Variance Estimation of Random Forests with Infinite-Order U-statistics." arXiv preprint arXiv:2202.09008.
- Formentini, S. E., Wei L., & Zhu, R. (2022) "Confidence Band Estimation for Survival Random Forests." arXiv preprint arXiv:2204.12038.

Examples

```
set.seed(42)
x <- matrix(rnorm(300 * 5), ncol = 5)
y <- rowSums(x[, 1:2]) + rnorm(300)
fit <- RLT(x, y, ntrees = 100)
print(fit)
```

Index

`cindex`, [2](#)

`forest.kernel`, [3](#)

`get.one.tree`, [4](#)

`get.surv.band`, [5](#)

`importance`, [6](#)

`predict.RLT`, [7](#)

`print.importance.RLT`, [9](#)

`print.RLT`, [9](#)

`RLT`, [6](#), [10](#)