# Package: PRTree (via r-universe)

May 20, 2026

**Type** Package

**Date** 2026-02-18

**Title** Probabilistic Regression Trees

**Version** 1.0.3

**Depends** R (>= 4.3.0)

**Description** Implementation of Probabilistic Regression Trees (PRTree), providing functions for model fitting and prediction, with specific adaptations to handle missing values. The main computations are implemented in 'Fortran' for high efficiency. The package is based on the PRTree methodology described in Alkhoury et al. (2020), ``Smooth and Consistent Probabilistic Regression Trees'' <https://proceedings.neurips.cc/paper_files/paper/2020/file/8289889263db4a40463e3f358bb7c7a1-Paper.pdf>. Details on the treatment of missing data and implementation aspects are presented in Prass, T.S.; Neimaier, A.S.; Pumi, G. (2025), ``Handling Missing Data in Probabilistic Regression Trees: Methods and Implementation in R'' <doi:10.48550/arXiv.2510.03634>.

**License** GPL (>= 3)

**Encoding** UTF-8

**NeedsCompilation** yes

**RoxygenNote** 7.3.3

**Author** Alisson Silva Neimaier [aut] (ORCID: <https://orcid.org/0000-0002-7524-0776>), Taiane Schaedler Prass [aut, ths, cre] (ORCID: <https://orcid.org/0000-0003-3136-909X>)

**Maintainer** Taiane Schaedler Prass <taianeprass@gmail.com>

**Repository** https://tsprass.r-universe.dev

**Date/Publication** 2026-02-18 22:30:02 UTC

**RemoteUrl** https://github.com/cran/PRTree

**RemoteRef** HEAD

**RemoteSha** c89de6e938b3c10f96cebffab6a36f7f9389a839

# Contents



---

`pr_tree` *Probabilistic Regression Trees (PRTrees)*

---

## Description

Fits a Probabilistic Regression Tree (PRTree) model. This is the main user-facing function of the package.

## Usage

```
pr_tree(y, X, control = pr_tree_control(), ...)
```

## Arguments

| | |
|---|---|
| `y` | A numeric vector for the dependent variable. |
| `X` | A numeric matrix or data frame for the independent variables. |
| `control` | A list of control parameters, typically created by ‘pr_tree_control()‘. Alternatively, control parameters can be passed directly via the ‘...‘ argument. |
| `...` | Control parameters to be passed to ‘pr_tree_control()‘. These will override any parameters specified in the ‘control‘ list. |

## Value

An object of class ‘prtree‘ containing the fitted model. This is a list with the following components

| | |
|---|---|
| `yhat` | The estimated values for ‘y‘. |
| `XRegion` | A matrix with two columns indicating the terminal node (region) each observation belongs to. The first column (‘TRUE‘) may have ‘NA‘ for observations with missing values. The second column (‘Internal‘) shows the region assigned by the algorithm. |
| `dist` | The Fortran code corresponding to the distribution used. (For prediction purposes) |
| `par_dist` | Parameters related to the distribution (if any). |
| `fill_type` | Fortran code corresponding to the method used to fill the matrix P when missing values are present. |
| `P` | The matrix of probabilities for each terminal node. |
| `gamma` | The values of the $\gamma_j$ weights estimated for the returned tree |
| `MSE` | The mean squared error for the training, test/validation, and global datasets. |

`sigma` The optimal $\sigma$ vector selected by the grid search.

`nodes_matrix_info`
A matrix with information for each node of the tree.

`regions` A data frame with the bounds of each variable in each node of the returned tree.

## Examples

```
set.seed(1234)
X <- matrix(runif(200, 0, 10), ncol = 1)
eps <- matrix(rnorm(200, 0, 0.05), ncol = 1)
y <- cos(X) + eps

# Fit model with custom controls passed directly
reg <- pr_tree(y, X, max_terminal_nodes = 9, perc_test = 0)

plot(
  X[order(X)], reg$yhat[order(X)],
  xlab = "x", ylab = "cos(x)", col = "blue", type = "l"
)
points(
  X[order(X)], y[order(X)],
  xlab = "x", ylab = "cos(x)", col = "red"
)
```

---

# `pr_tree_control` *Set Control Parameters for PRTree*

---

## Description

This function creates a list of control parameters for the ‘pr_tree‘ function, with validation for each parameter.

## Usage

```
pr_tree_control(sigma_grid = NULL, grid_size = 8,
  max_terminal_nodes = 15L, cp = 0.01, max_depth = max_terminal_nodes -
  1, n_min = 5L, perc_x = 0.1, p_min = 0.05, perc_test = 0.2,
  idx_train = NULL, fill_type = 2L, proxy_crit = "both",
  n_candidates = 3L, by_node = FALSE, dist = "norm", iprint = -1, ...)
```

## Arguments

`sigma_grid` Optional, a numeric value, vector or a matrix with candidate values for the parameter $\boldsymbol{\sigma}$, to be passed to the grid search algorithm. If a single numeric value is provided, the code assumes that $\sigma_j = \sigma$ for all $j$ covariates. If `NULL`, the standard deviations of the columns in `X` are used to create a grid with values in the interval $(0, 2\hat{\sigma}_j]$, with increments of $\hat{\sigma}_j/4$, where $\hat{\sigma}_j$ denotes the standard deviation of the $j$th covariate. The default is `NULL`.

`grid_size` Optional, the number of candidate values for ‘sigma‘ to generate when ‘sigma_grid‘ is ‘NULL‘. Default is 8.

`max_terminal_nodes`
A non-negative integer. The maximum number of regions in the output tree. The default is 15.

`cp` A positive numeric value. The complexity parameter. Any split that does not decrease the MSE by a factor of ‘cp‘ will be ignored. The default is 0.01.

`max_depth` A non-negative integer. The maximum depth of the decision tree. The depth is defined as the length of the longest path from the root to a leaf. The default is 14.

`n_min` A positive integer, The minimum number of observations in a final node. The default is ‘max_terminal_nodes - 1‘.

`perc_x` A positive numeric value between 0 and 1. Given any column of $P$, ‘perc_x‘ is the minimum proportion of rows that must have a probability higher than ‘p_min‘ for a splitting attempt to be made in the corresponding region. The split will be ignored if any of the resulting regions do not meet the same criterion. The default is 0.1.

`p_min` A positive numeric value. A threshold probability that controls the splitting process. A splitting attempt is made in a given region only when the proportion of rows with probability higher than ‘p_min‘, in the corresponding column of the matrix $P$, is equal to `perc_x`. The default is 0.05.

`perc_test` A numeric value between 0 (inclusive) and 1 (exclusive) that specifies the proportion of the data to be held out for model validation or testing. Default is 0.2. The role of this hold-out set depends on the ‘sigma_grid‘

- **Validation Set:** If ‘sigma_grid‘ contains multiple candidate $\sigma$ values (‘grid_size > 1‘), ‘perc_test‘ of the data is used as a validation set to select the best $\sigma$ based on out-of-sample Mean Squared Error (MSE). If ‘perc_test‘ is 0, ‘sigma‘ will be selected based on the MSE for the training sample.
- **Test Set:** If a single, fixed $\sigma$ is provided (‘grid_size = 1‘), ‘perc_test‘ of the data is used as a test set to evaluate the final model's performance. If ‘perc_test‘ is 0, the entire dataset is used for training.

The data split is performed using stratified sampling to ensure that the proportion of observations with missing values is similar across the training and validation/test sets.

`idx_train` Indexes for the training sample. Default is ‘NULL‘, in which case the indexes are computed based on the ‘perc_test‘ argument. If ‘idx_train‘ is provided, ‘perc_test‘ is ignored.

`fill_type` Integer indicating the method to be used to fill the probability matrix when ‘X‘ has NA's. Default is 2.

- ‘0‘: uniform (same probability for both child nodes).
- ‘1‘: attributes all probability to the child node that is compatible with the observed values.
- ‘2‘: computes the probability restricted to the observed entries

`proxy_crit` Character. Default is ‘"both"‘. Criterion used to associate an observation with missing values to a region:

- ‘"mean"‘: maximizes the difference in means after a split.
- ‘"var"‘: maximizes the variability between nodes.
- ‘"both"‘: combines the ‘"mean"‘ and ‘"var"‘ criteria.

`n_candidates` Integer. The number of competing candidates to consider when searching for the best split. To select the candidates, a proxy improvement measure is used. Then a full analysis is performed to choose the best among the ‘n_candidates‘ candidates. Default is 3.

`by_node` Logical. If ‘TRUE‘, the algorithm selects ‘n_candidates‘ for each node and then makes a full analysis to choose the best among all nodes. Otherwise the ‘n_candidates‘ are selected globally. Default is ‘FALSE‘.

`dist` Character. The distribution to be used in the model. One of ‘"norm"‘ (Gaussian), ‘"lnorm"‘ (log-normal), ‘"t"‘ (Student's $t$), or ‘"gamma"‘ (Gamma). Default is ‘"norm"‘.

`iprint` Integer. Controls the verbosity of the Fortran backend. Default is -1 (silent).

- ‘iprint < 0‘: No printing.
- ‘iprint = 0‘: Prints basic information.
- ‘iprint > 0‘: As for ‘iprint = 0‘ plus progress reports.

`...` Extra parameters to be passed to the chosen distribution.

- ‘"norm"‘: Uses the standard Gaussian distribution. No extra parameters required.
- ‘"lnorm"‘: Uses the log-normal distribution with ‘meanlog = 0‘. Requires ‘sdlog‘.
- ‘"t"‘: Uses the $t$ distribution. Requires ‘df‘.
- ‘"gamma"‘: Uses the gamma distribution with ‘scale = 1‘. Requires ‘shape‘.

## Value

A list of class ‘prtree.control‘ containing the validated control parameters.

## Examples

```
# Get default control parameters
controls <- pr_tree_control()

# Customize some parameters
custom_controls <- pr_tree_control(max_depth = 5, n_candidates = 5)
```

---

`predict.prtree` *Predict from a Probabilistic Regression Tree Model*

---

## Description

Obtains predictions from a fitted ‘prtree‘ object.

## Usage

```
## S3 method for class 'prtree'
predict(object, newdata, complete = FALSE, ...)
```

## Arguments

| | |
|---|---|
| object | An object of class ‘prtree‘, as returned by ‘pr_tree()‘. |
| newdata | A data frame or matrix containing new data for which to generate predictions. Must contain the same predictor variables as the data used to fit the model. |
| complete | Logical. If ‘FALSE‘ (default), only the vector of predicted values is returned. If ‘TRUE‘, a list containing both the predicted values and the probability matrix ‘P‘ is returned. |
| ... | further arguments passed to or from other methods. |

## Value

If ‘complete = FALSE‘, a numeric vector of predicted values (‘yhat‘). If ‘complete = TRUE‘, a list containing:

| | |
|---|---|
| yhat | The numeric vector of predicted values. |
| P | The probability matrix for the new data. |

# Index