Package: glmtree (via r-universe)

October 12, 2026

Type Package

Title Logistic Regression Trees

Version 0.4.0

Date 2026-10-11

Maintainer Adrien Ehrhardt <adrien.ehrhardt@polytechnique.edu>

Description A logistic regression tree is a decision tree with logistic regressions at its leaves. A particular stochastic expectation maximization algorithm is used to draw a few good trees, that are then assessed via the user's criterion of choice among BIC / AIC / test set Gini. The formal development is given in a PhD chapter, see Ehrhardt (2019) <https://github.com/adimajo/manuscrit_these/releases/>.

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 4.1.0)

Imports graphics, methods, partykit, stats

Suggests FactoMineR, knitr, rmarkdown, testthat (>= 3.0.0)

URL https://adimajo.github.io/glmtree/, https://github.com/adimajo/glmtree

BugReports https://github.com/adimajo/glmtree/issues

VignetteBuilder knitr

Config/testthat/edition 3

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.1.1

Collate 'allClasses.R' 'cutDataset.R' 'generateData.R' 'glmtree.R' 'helpers.R' 'initialization.R' 'normalizedGini.R' 'predict.R'

Repository https://adimajo.r-universe.dev

Date/Publication 2026-10-12 19:09:54 UTC

RemoteUrl https://github.com/adimajo/glmtree

RemoteRef HEAD

RemoteSha 429bfd53db9a00b49348e89df2032092fe1f634e

Contents

generateData 2
glmtree 3
glmtree-class 4
normalizedGini 5
predict 6

Index **7**

`generateData` *Generates data from two logistic regression trees.*

Description

This function generates data from two logistic regression trees: one with three apparent clusters (in terms of variance of the features) but a single logistic regression generating y | x, and one with a single apparent cluster but three different logistic regressions generating y | x given a categorical feature.

Usage

```
generateData(n = 100, scenario = "tree", visualize = FALSE)
```

Arguments

`n`	The number of observations to draw.
`scenario`	The "no tree" scenario denotes the first scenario where there is a single logistic regression generating the data. The "tree" scenario generates data from the second data generating mechanism where there are three logistic regressions.
`visualize`	Whether (TRUE) or not (FALSE) to plot the generated data.

Value

Generates and returns data according to a true logistic regression tree (if scenario = "tree") or a single regression tree (if scenario = "no tree"). Eventually plots this dataset (if visualize = TRUE).

Author(s)

Adrien Ehrhardt

Examples

```
generateData(scenario = "tree")
```

glmtree *Logistic regression tree by Stochastic-Expectation-Maximization*

Description

This function produces a logistic regression tree: a decision tree with logistic regressions at its leaves. The segmentation of the observations is a latent variable: at each iteration a logistic regression is fitted in each segment, a conditional inference tree (`partykit::ctree`) is fitted to the current segmentation, and each observation is re-assigned to a segment drawn proportionally to the tree probability times the likelihood of its label under the segment's regression. A few good trees are kept along the chain and the best one is chosen according to `criterion`.

Usage

```
glmtree(
  x,
  y,
  K = 10,
  iterations = 200,
  test = FALSE,
  validation = FALSE,
  proportions = c(0.3),
  criterion = "bic",
 ctree_controls = partykit::ctree_control(alpha = 0.1, minbucket = 100, maxdepth = 5),
  burn_in = min(50L, iterations%/%4L),
  patience = NULL,
  verbose = FALSE
)
```

Arguments

`x`	The features to use for prediction (data frame or matrix with numeric, integer and factor columns; characters and logicals are converted to factors). No missing values are allowed.
`y`	The binary / boolean labels to predict. The larger value (or the second level of a factor) is the positive class.
`K`	The number of segments to start with (maximum number of segments there'll be in the end).
`iterations`	The number of iterations to do in the SEM protocole (default: 200).
`test`	Boolean : True if a test set should be held out, on which the `criterion` of the best tree is reported in `performance` (default: FALSE).
`validation`	Boolean : True if a validation set should be held out, on which the `criterion` is computed at each iteration to choose the best tree (default: FALSE, the criterion is then computed on the training set).

proportions	The proportions of observations held out: `c(test, validation)` when both are requested, otherwise a single proportion for whichever is requested. Not used when both are FALSE. Errors when invalid (e.g. a sum greater than one). Default: 0.3.
criterion	The criterion ('gini','aic','bic') used to choose the best tree among the generated ones (default: 'bic'). AIC and BIC are summed over segments (the BIC penalty of a segment uses the logarithm of its size) and are replaced by the log-likelihood when computed on held out data, where no penalty is needed. The Gini index is best used with a validation set.
ctree_controls	The controls to use for `partykit::ctree`.
burn_in	The number of first iterations that cannot be chosen as the best tree (default: the minimum of 50 and a quarter of `iterations`).
patience	If not `NULL`, the chain stops when the criterion did not improve for `patience` iterations (after the burn-in).
verbose	Whether to print the criterion at each iteration.

Value

An S4 object of class `glmtree-class` that contains the parameters used to search for the logistic regression tree, the best tree it found, and its performance. Use `predict` to predict on new data.

Author(s)

Adrien Ehrhardt

Examples

```
set.seed(1)
data <- generateData(n = 300, scenario = "tree")
fit <- glmtree(
  x = data[, c("x1", "x2", "x3")], y = data$y, K = 5, iterations = 20,
  ctree_controls = partykit::ctree_control(alpha = 0.1, minbucket = 30, maxdepth = 3)
)
fit
head(predict(fit, data[, c("x1", "x2", "x3")]))
```

`glmtree-class` *Class glmtree*

Description

Class `glmtree` represents a logistic regression tree scheme associated with its optimal logistic regression models.

Usage

```
## S4 method for signature 'glmtree'
show(object)

## S4 method for signature 'glmtree'
print(x, ...)
```

Arguments

`object`	The S4 `glmtree` object.
`x`	The S4 `glmtree` object.
`...`	Not used.

Slots

`parameters` The parameters associated with the method.

`best.tree` The best tree found by the method given its parameters: `tree` (the `partykit::ctree` giving the segment of an observation, `NULL` if the chain collapsed to a single segment), `glms` (the logistic regression of each segment, a named list of `glm` objects) and the elements needed to predict on new data.

`performance` The performance obtained with the method given its parameters: `performance` (the criterion of the best iteration, computed on the test set if there is one), `criterionEvolution` (the criterion at each iteration) and `best_iteration`.

`normalizedGini` *Calculating the normalized Gini index*

Description

This function calculates the Gini index of a classification rule outputting probabilities. It is a classical metric in the context of Credit Scoring. It is equal to 2 times the AUC (Area Under ROC Curve) minus 1, computed from the ranks of the predictions so that the result does not depend on the order of the observations when predictions are tied.

Usage

```
normalizedGini(actual, predicted)
```

Arguments

`actual`	The numeric binary vector of the actual labels observed.
`predicted`	The vector of the probabilities predicted by the classification rule.

Value

The Gini index of the predicted probabilities as a single `num` (`NA` if `actual` has a single class).

Author(s)

Adrien Ehrhardt

Examples

```
normalizedGini(c(1, 1, 1, 0, 0), c(0.7, 0.9, 0.5, 0.6, 0.3))
```

---

predict — *Prediction of a logistic regression tree*

Description

This defines the method "predict" for objects of S4 class `glmtree-class`.

Usage

```
predict(object, ...)

## S4 method for signature 'glmtree'
predict(object, newdata, type = c("response", "segment"), ...)
```

Arguments

`object`	The S4 `glmtree` object.
`...`	Not used.
`newdata`	A data frame with the features used to fit the tree.
`type`	`"response"` for the predicted probability of the positive class, `"segment"` for the segment of each observation.

Details

Applies the best tree to new observations to find their segment, then the logistic regression of that segment. Observations with a factor level not seen in training receive the rate of positives of their segment.

Value

A numeric vector with one prediction per row of `newdata`.

Examples

```
set.seed(1)
data <- generateData(n = 300, scenario = "tree")
fit <- glmtree(
  x = data[, c("x1", "x2", "x3")], y = data$y, K = 5, iterations = 20,
  ctree_controls = partykit::ctree_control(alpha = 0.1, minbucket = 30, maxdepth = 3)
)
table(predict(fit, data[, c("x1", "x2", "x3")], type = "segment"))
```

Index

∗ **gini**
    normalizedGini, 5
∗ **glmtree**
    generateData, 2
    glmtree, 3
∗ **index**
    normalizedGini, 5
∗ **predict prediction**
    predict, 6

generateData, 2
glmtree, 3
glmtree-class, 4

normalizedGini, 5

predict, *4*, 6
predict,glmtree-method *(*predict*)*, 6
print,glmtree-method *(*glmtree-class*)*, 4

show,glmtree-method *(*glmtree-class*)*, 4