Package ‘SplitSoftening’

October 12, 2022

Encoding UTF-8

Version 2.1-0

Title Softening Splits in Decision Trees

Author Jakub Dvorak

Maintainer Jakub Dvorak <JakubDvorak@email.cz>

Depends R (>= 3.0.0)

Enhances tree

Suggests gsl

Description Allows to produce and use classification trees with soft (probability) splits, as described in: Dvořák, J. (2019), <doi:10.1007/s00180-019-00867-1>.

License GPL (>= 2)

RoxygenNote 7.1.2

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-10-08 10:40:02 UTC

R topics documented:

`predictSoftsplits` *Prediction according to 'soft tree'.*

Description

Prediction according to 'soft tree'.

Usage

```
predictSoftsplits(fit, newdata)
```

Arguments

`fit`	The soft tree.
`newdata`	Data to classify.

Value

The matrix of predicted class probabilities.

`soften` *Create a 'soft tree' structure with softening parameters set using one of the named method.*

Description

This is a convenience method implemented over `softsplits` and the softening functions from this package.

Usage

```
soften(fit, ds, method, control = NULL)
```

Arguments

`fit`	A classification tree - either an object `tree` which represents a classification tree, or already a 'soft tree' like created by the `softsplits` function.
`ds`	A data frame used as training data for softening
`method`	A name of softening method. One of: "DR0", "DR1", "DR2", ..., "ESD", "C4.5", "optim_d", "optim_d^2", "optim_d^4", "optim_auc" The 'method = "DR*x*"' for some number *x*: The softening parameters are set according to 'data ranges' appropriate to tree nodes. The parameters are configured such that in each node the distance of the boundary of the softened area

from split value is $2^{-x}r$, where r is the distance from the split value to the furthest data point in the tree node projected to the direction from the split value to the boundary.

The 'method = "ESD"' sets boundaries of the softening using error standard deviation. This is how C4.5 method sets "probabilistic splits"; for that reason value `"C4.5"` is an alias for `"ESD"`.

The 'method = "optim_d^q"' for some number q: The softening parameters are set by optimization process which minimizes `mean`$((1.0 - p)^q)$ where p is for each data point in `ds` the predicted probability of the correct label.

If 'method = "optim_auc"': The classification tree `fit` must perform prediction to two classes. The value of the 'area under ROC curve' computed on the data set `ds` is maximized by optimization.

`control` List of additional configuration paramaters. Possible members in the list are: `verbosity`, `implementation`, `iteration.count`, `sft.ini`, which correspond to the paramaters of `softening.optimized`.

Value

The 'soft tree' structure representing the same tree structure as given in the parameter `fit`, but with softening parameters set using the given method.

See Also

`predictSoftsplits`.

softening.by.data.range

Make split softening based on data ranges.

Description

This softening configures each softening parameter in the tree according to 'data ranges' appropriate to tree nodes. The parameters are configured such that in each node the distance of the boundary of the softened area from split value is `factor` * the distance from the split value to the furthest data point in the tree node projected to the direction from the split value to the boundary.

Usage

```
softening.by.data.range(tr, ds, factor = 1)
```

Arguments

`tr` The soft tree

`ds` The data set to be used for determining data boundaries

`factor` The scalar factor

Value

The soft tree with the new softening parameters

Examples

```
if(require(tree)) {
  train.data <- iris[c(TRUE,FALSE),]
  test.data <- iris[c(FALSE,TRUE),]
  tr <- tree( Species~., train.data )

  # tree with ”zero softening”
  s0 <- softsplits( tr )
  # softened tree
  s1 <- softening.by.data.range( s0, train.data, .5 )

  response0 <- predictSoftsplits( s0, test.data )
  response1 <- predictSoftsplits( s1, test.data )
  # get class with the highest response
  classification0 <- levels(train.data$Species)[apply( response0, 1, which.max )]
  classification1 <- levels(train.data$Species)[apply( response1, 1, which.max )]

  # compare classifiction to the labels
  table( classification0, test.data$Species )
  table( classification1, test.data$Species )
}
```

`softening.by.esd` *Soften splits separately using error standard deviation.*

Description

Set boundaries determined by given data to the splits in the tree, such that in any inner node if its splitting value would be moved there, then the number of misclassified cases in this node would be one standard deviation over the actual misclassification.

Usage

```
softening.by.esd(fit, d)
```

Arguments

`fit`	The soft tree
`d`	The data set

Details

This is the same approach as C4.5 uses for "probabilistic splits"

References

Quinlan, J. Ross (1993), *C4.5: programs for machine learning*, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

softening.optimized *Make split softening optimized with Nelder-Mead.*

Description

This softening configures all parameters in the tree with optimization method Nelder-Mead to minimize the given ‘miss’ function.

Usage

```
softening.optimized(
  tr,
  d,
  miss.fn,
  verbosity = 0,
  implementation = c("gsl", "R"),
  iteration.count = NULL,
  sft.ini = 1
)
```

Arguments

tr	The soft tree
d	The data set to be used in intialization for determining data boundaries and in optimization step to evaluate the objective function on the predictions on this data set by the soft tree with updated softening parameters.
miss.fn	Function to provide the value of the objective function for optimization. The function obtains as an argument the matrix of class probabilities as returned by predictSoftsplits when making predictions for the data set d using the soft tree tr but with some softening parameters reset within optimization procedure. The function is expected to return one numeric value; this value is minimized by the optimization method.
verbosity	The verbosity level configures how many additional information is printed
implementation	Indentify implementation of optimizer.
iteration.count	Number of optimizer iterations.
sft.ini	Parameter of softening used as the initial value for the optimization.

- `"gsl"` uses `multimin` function from `gsl` package. Note: In the current version (2.1-6) of `gsl` package this function does not work.
- `"R"` uses `optim` - the standard optimization function in R.

Value

The soft tree with the new softening parameters

`softsplits` *Create 'soft tree' structure from a tree object.*

Description

Create 'soft tree' structure from a tree object.

Usage

```
softsplits(fit)
```

Arguments

`fit`	A tree object: must be a classification tree

Value

A data structure suitable for softening splits in the tree and for evaluation of 'soft tree' on submitted data. The returned object is ready for softening, but it is not yet softened. The result of prediction for some data with the returned object is still the same as with the original tree `fit`.

See Also

`predictSoftsplits`.

`SplitSoftening` *Package: Softening splits in classification trees*

Description

The basic idea of split softening is to modify the process of classification of an input case with a decision tree such that in the area near the threshold of a softened split both branches of the tree are used to provide a prediction for the submitted case and their results are combined.

Details

Functions in this package allow to add softening to the nodes of a classification tree created with the package `tree`. Each node where a decision on a continuous variable is made is enriched with softening parameters which specify the boundaries of the softening area and which together with the original split threshold determine the weights of the branches when combined.

The weights of branches are (1/2, 1/2) in the original split threshold. Other points inside the softening area have weights given by linear interpolation to reach the values (0, 1), or vice versa, on the boundaries of the softening area.

A data structure for a decision tree prepared for softening can be created from a `tree` object with the `softsplits` function.

Softening parameters may be set to the 'soft tree' structure. The package offers the following functions for this purpose:

- `softening.by.data.range`
- `softening.by.esd`
- `softening.optimized`
- `soften`

A softened tree might be used to obtain a prediction for a dataset using the `predictSoftsplits` function.

References

Dvořák, J. (2019), *Classification trees with soft splits optimized for ranking* <doi:10.1007/s00180-019-00867-1> `https://rdcu.be/bkeW2`

Index