Package 'climbeR'

November 19, 2016

Title Calculate Average Minimal Depth of a Maximal Subtree for 'ranger' Package Forests

Version 0.0.1

Description Calculates first, and second order, average minimal depth of a maximal subtree for a forest object produced by the R 'ranger' package. This variable importance metric is implemented as described in Ishwaran et. al. (``High-Dimensional Variable Selection for Survival Data", March 2010, <doi:10.1198/jasa.2009.tm08622>).

Depends R (>= 3.3.2)

License MIT + file LICENSE

Imports ggplot2, grDevices, graphics

Suggests survival (>= 2.38-1), ranger, knitr, rmarkdown, testthat, dplyr

RoxygenNote 5.0.1

VignetteBuilder knitr

NeedsCompilation no

Author John Karlen [aut, cre]

Maintainer John Karlen <jkarlen@massmutual.com>

Repository CRAN

Date/Publication 2016-11-19 10:13:21

R topics documented:

`baseRPlotting` *Plot Second Order vs First Order minimal depth of a maximal subtree depth for base R*

Description

Base r version of plotFirstAndSecondOrderMetric (just an alternative).

Usage

```
baseRPlotting(eval_data)
```

Arguments

`eval_data`	The output of calculateAMDMS.

Value

plot object

`binForestByDepth` *Bin Forest by Depth*

Description

Given a forest object from the ranger package, this function will bin the forest into depths. This is a helper function for the 'calculateAMDMS' function.

Usage

```
binForestByDepth(ranger_obj)
```

Arguments

`ranger_obj`	A ranger object from the ranger package, which was created with param write.forest set to TRUE. In other words, it must have a 'forest' property.

Value

A list with 3 elements. The first is a list of vectors - one for each independent variable ocurring in the forest (this may not be the complete set of independent variables, but we will account for any variables that do not occur in the forest later). Each vector contains all minimal depths of maximal subtrees in the forest, for the corresponding independent variable. The second element is a vector of tree heights 'forest_depths'. The third element is a set of variable id's for matching to independent variable names.

calculateAMDMS *Forest Averaged Minimal Depth of a Maximal Subtree (AMDMS)*

Description

Given a result from the Ranger package (write.forest must be set to TRUE), this function will traverse the trees and calculate the first and second order average minimal depth of a maximal subtree.

Usage

```
calculateAMDMS(ranger_obj)
```

Arguments

ranger_obj	A ranger object from the ranger package, which was created with param write.forest set to TRUE. In other words, it must have a 'forest' property.

Value

A data.frame with two columns: averaged first and second order minimal depth of a maximal subtree.

countSplitsPerVar *Count Splits Per Variable*

Description

This function counts the number of times each variable was used to split a tree.

Usage

```
countSplitsPerVar(ranger_obj_forest)
```

Arguments

ranger_obj_forest	A ranger object from the ranger package, which was created with param write.forest set to TRUE. In other words, it must have a 'forest' property.

Value

A dataframe with one column of counts, and one column of normalized counts. Rows are labeled by variable names.

getAndPlotMetric *Calculate the metric and make Second Order vs First Order plot*

Description

Comprehensive function for calculating minimal depth of a maximal subtree averaged over the forest and then plotting the result.

Usage

```
getAndPlotMetric(ranger_result, plot_missing_so = FALSE)
```

Arguments

ranger_result	A ranger object from the ranger package, which was created setting param write.forest to TRUE. In other words, it must have a 'forest' property.
plot_missing_so	An optional parameter to show features that only have a first order metric value. Variables can have high feature strength, but may be unlikely to have a second maximal subtree because of low cardinality.

Value

a list; element 1 is a data.frame containing subtree depth data, element 2 is the plot of number of splits vs first order metric, element 3 is a plot of second order vs first order.

Examples

```
require(survival)
library(ranger)
rg.veteran <- ranger(Surv(time, status) ~ ., data = veteran, write.forest =
TRUE)
result <- getAndPlotMetric(rg.veteran)
```

lookForVarsAbsentInForest *Look for Variable ID's that didn't occur in the Forest.*

Description

Find any remaining vars, if missing. Vars can be absent in the forest, if they were never used to split. This function does some bookkeeping, to find elements in the count vector that should be 0. If there weren't enough vars observed, their indeces must be either at the end of vars_used, or the beginning.

Usage

```
lookForVarsAbsentInForest(counts, vars_used, num_ind_vars, forest)
```

Arguments

counts	A vector of split counts in the forest. This may need to be updated with 0's for variables that didn't occur in the forest.
vars_used	The current list of varID's that have been found in the forest.
num_ind_vars	The number of independent vars. Counts must have this many elements.
forest	Pass this to access the 'status.varID' if necessary.

Value

updated counts vector.

---

plotAMDMSvsNumSplits *Plot metric vs number of splits per variable.*

Description

Plots the metric against the number of splits in the forest. Features with many splits will be weighed unfairly by the metric.

Usage

```
plotAMDMSvsNumSplits(eval_data, add_text_labels = FALSE)
```

Arguments

eval_data	The output of calculateAMDMS.
add_text_labels	An optional parameter to turn on text labels next to the feature's dot on the plot.

Value

plot object

`plotFirstAndSecondOrderMetric` *Plot Second Order vs First Order, minimal depth of a maximal subtree, averaged over the forest (shortened to "metric" for brevity)*

Description

Given evaluated data from calculateAMDMS, plot the result.

Usage

```
plotFirstAndSecondOrderMetric(eval_data, plot_missing_so = FALSE,
  add_text_labels = FALSE)
```

Arguments

`eval_data`	The output of calculateAMDMS.
`plot_missing_so`	An optional parameter to show features that only have a first order metric value. Variables can have high feature strength, but may be unlikely to have a second maximal subtree because of low cardinality.
`add_text_labels`	An optional parameter to turn on text labels next to the feature's dot on the plot.

Value

so_vs_fo A ggplot2 object, which shows second order vs first order average minimal depth of a maximal subtree depth.

`recursiveDepthBinning` *Recursive Depth Binning*

Description

Function to recursively traverse depths of a tree.

Usage

```
recursiveDepthBinning(node_list, depth, expected_num_children, binned_depths)
```

Arguments

node_list	Must be in the format of elements in the Ranger package's forest$split.varIDs, which represents one tree in the forest. Recursion is done by counting the number of terminal nodes at the current depth to anticipate the correct number of nodes at the next depth.
depth	Each recursive call must know the current depth in the tree.
expected_num_children	The number of nodes in the current depth must be anticipated, given the number of terminal nodes at the previous depth.
binned_depths	A list passed between recursive calls, to store results.

Value

A list of vectors, where elements correspond to depths, and vectors contain variable ID's of variables used to split at that depth.

---

startRecursiveDepthBinning

Start Recursive Depth Binning

Description

The starter function for the recursion in recursiveDepthBinning.

Usage

```
startRecursiveDepthBinning(tree_split_varIDs)
```

Arguments

tree_split_varIDs	Given one element of a 'split.varIDs' list, this function will pass it to the recursiveDepthBinning function to bin the tree by depth, starting at the root.

Value

A list with an element per depth encountered. Each element is a vector of variable IDs

Index