Package 'vistla'

November 24, 2022

Title Detecting Influence Paths with Information Theory

Version 1.0.1

Description Traces information spread through interactions between features, utilising information theory measures and a higher-order generalisation of the concept of widest paths in graphs. In particular, 'vistla' can be used to better understand the results of high-throughput biomedical experiments, by organising the effects of the investigated intervention in a tree-like hierarchy from direct to indirect ones, following the plausible information relay circuits.
Due to its higher-order nature, 'vistla' can handle multimodality and assign multiple roles to a single feature.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.2

Depends R (>= 3.5.0)

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Miron B. Kursa [aut, cre] (<https://orcid.org/0000-0001-7672-648X>)

Maintainer Miron B. Kursa <m@mbq.me>

Repository CRAN

Date/Publication 2022-11-23 23:20:02 UTC

R topics documented:

`agreement` *Measure agreement between vistla trees*

Description

Measure agreement between vistla trees

Usage

```
agreement(x, y = NULL, ..., method = "spearman", raw = FALSE)
```

Arguments

`x`	a `vistla` object (first to compare) or a list of `vistla` objects (all to compare pairwise).
`y`	a `vistla` object, second to compare, if `x` is a single object.
`...`	ignored.
`method`	correlation method to use for quantification. See `cor` for possible values.
`raw`	if `TRUE`, suppresses correlation calculation and output the raw aligned scores instead.

Value

Correlation matrix with score correlations between each pair of given vistla trees.

Examples

```
data(chain)
agreement(
 vistla(Y~.,data=chain),
 vistla(Y~.,data=chain[,sample(6)])
)
```

branches *Extract all branches of the Vistla tree*

Description

Gives access to a list of all branches in the tree.

Usage

```
branches(x, suboptimal = FALSE)

## S3 method for class 'vistla'
as.data.frame(x, row.names = NULL, optional = FALSE, suboptimal = FALSE, ...)
```

Arguments

x	vistla object.
suboptimal	if TRUE, sub-optimal branches are included.
row.names	passed to as.data.frame.
optional	passed to as.data.frame.
...	ignored.

Value

A data frame collecting all branches traced by vistla. Each row corresponds to a single branch, i.e., edge between feature pairs. This way it is a triplet of original features, names of which are stored in a, b and c columns. For instance, path $I \to J \to K \to L \to M$ would be stored in three rows, for (a, b, c)=(I, J, K), (J, K, L) and (K, L, M). The width of a path (minimal ι value) between root and feature pair (b, c) is stored in the score column. depth stores the path depth, starting from 1 for pairs directly connected to the root, and increasing by one for each additional feature. Final column, leaf, is a logical path indicating whether the edge is a final segment of the widest path between root and c.

Note

Pruned trees (obatined with prune and using targets argument in the vistla call) have no suboptimal branches.

chain *Synthetic data representing a simple mediator chain*

Description

Chain is generated from a simple Bayes network,

$$X \to M_1 \to M_2 \to M_3 \to M_4 \to Y$$

where every variable is binary. The set consists of 11 observations, and is tuned to be easily deciphered.

Usage

```
data(chain)
```

Format

A data set with six columns, each is a factor of two levels.

hierarchy *Extract the vertex hierarchy from the vistla tree*

Description

Traverses the vistla tree in a depth-first order and lists the visited vertices as a data frame.

Usage

```
hierarchy(x)
```

Arguments

x	vistla object.

Value

A data frame of a class `vistla_hierarchy`.

Note

This function effectively prunes the tree off suboptimal paths.

junction *Synthetic data representing a junction*

Description

Junction is a model of a multimodal agent, a variable that is an element of multiple separate paths. Here, these paths are $A_1 \to X \to A_2$ and $B_1 \to X \to B_2$, while X is the junction. The set consists of 12 observations, and is tuned to be easily deciphered.

Usage

```
data(junction)
```

Format

A data set with five columns, each is a factor of two or four levels.

leaf_scores *Extract leaf scores of vertex pairs*

Description

Produces a matrix S where S_{ij} is a score of the path ending in vertices i and j. Since vistla works on vertex pairs, this value is unique. This can be interpreted as a feature similarity matrix in context of the current vistla root.

Usage

```
leaf_scores(x)
```

Arguments

x	vistla object.

Value

A square matrix with leaf scores of all feature pairs.

Note

This function should be called on an unpruned vistla tree, otherwise the result will be mostly composed of zeroes.

mi_scores *Extract mutual information score matrix*

Description

Produces a matrix S where S_{ij} is a value of $I(X_i; X_j)$. This matrix is always calculated as an initial step of the vistla algorithm and stored in the vistla object.

Usage

```
mi_scores(x)
```

Arguments

x	vistla object.

Value

A symmetic square matrix with mutual information scores between features and root.

paths *List all paths*

Description

Executes path_to for all path possible targets and returns a list with the results.

Usage

```
paths(x, targets_only = !is.null(x$targets), detailed = FALSE)
```

Arguments

x	vistla object.
targets_only	if TRUE, only paths to targets are extracted. By default, turned on when x has targets, and off otherwise.
detailed	passed to path_to. If TRUE, suppresses default output and presents the same paths in a form of data frames featuring score.

Value

A named list with one element per leaf or target, containing the path between this feature and root, in a format identical to this used by the path_to function.

path_to *Extract a single path*

Description

Gives access to a vector of feature names over a path to a certain target feature.

Usage

```
path_to(x, target, detailed = FALSE)
```

Arguments

x	vistla object.
target	target feature name.
detailed	if TRUE, suppresses default output and presents the same paths as a data frame featuring score.

Value

By default, a character vector with names of features along the path from `target` into root. When `detailed` is set to `TRUE`, a `data.frame` in a format identical to this produced by `branches`, yet without the `leaf` column.

plot.vistla *Overview plot of the vistla tree*

Description

Overview plot of the vistla tree

Usage

```
## S3 method for class 'vistla'
plot(x, ..., scale_width = TRUE)
```

Arguments

x	vistla object.
...	additional graphical parameters, passed to plot.
scale_width	if TRUE, widths of links are scaled according to score.

Value

x, invisibly.

print.vistla_hierarchy *Print vistla objects*

Description

Utility functions to print vistla objects.

Usage

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

## S3 method for class 'vistla'
print(x, n = 7L, ...)
```

Arguments

x	vistla object.
...	ignored.
n	maximal number of paths to preview.

Value

Invisible copy of x.

prune *Prune the vistla tree*

Description

This function allows to filter out suboptimal branches, as well as weak ones or these not in particular paths of interest.

Usage

```
prune(x, targets, iomin)
```

Arguments

x	vistla object.
targets	a character vector of features. When not missing, all branches not on lying paths to these targets are pruned. Unreachable targets are ignored, while names not present in the analysed set cause an error.
iomin	a single numerical value. When given, it effectively overrides the value of iomin given to the vistla invocation; to this end, it can only be higher then the original value, since prune only modifies the output and cannot re-run the pathfinding.

Value

Pruned x; if both arguments are missing, this function still removes suboptimal branches.

Examples

```
data(chain)
v<-vistla(Y~.,data=chain)
print(v)
print(prune(v,targets="M3"))
print(prune(v,iomin=0.3))
```

---

vistla *Influence path identification with the Vistla algorithm*

Description

Detects influence paths.

Usage

```
vistla(x, ...)

## S3 method for class 'formula'
vistla(formula, data, ..., yn)

## S3 method for class 'data.frame'
vistla(
  x,
  y,
  ...,
  flow = c("fromdown", "intoup", "both", "spread", "from", "into", "up", "down"),
  iomin = 0,
  targets,
  verbose = FALSE,
  yn = "Y",
  threads = 0L
)

## Default S3 method:
vistla(x, ...)
```

Arguments

x	data frame of predictors.
...	pass-through arguments, ignored.

formula	alternatively, formula describing the task, in a form root~predictors, which adheres to standard R behaviours. Accepts + to add a predictor, - to omit one, and . to import whole data. Use I to calculate new predictors. When present in data, response is getting omitted from predictors.
data	data.frame in context of which the formula will be executed; can be omitted when not using ..
yn	name of the root (Y value), used in result pretty-printing and plots. Must be a single-element character vector.
y	vistla tree root, a feature from which influence paths will be traced.
flow	algorithm mode, specifying the iota function which gives local score to an edge of an edge graph. If in doubt, use the default, "fromdown".
iomin	score threshold below which path is not considered further. The higher value the less paths are generated, which also lowers the time taken by the function. The default value of 0 turns of this filtering. The same effect can be later achieved with the prune function.
targets	a vector of target feature names. If given, the algorithm will stop just after reaching the last of them, rather than after tracing all paths from the root. The same effect can be later achieved with the prune function.
verbose	when set to TRUE, turns on reporting of the algorithm progress.
threads	number of threads to use. Value of 0 indicates all available for OpenMP.

Value

The tracing results represented as an object of a class vistla. Use paths and path_to functions to extract individual paths, branches to get the whole tree and mi_scores to get the basic score matrix.

---

vistla_coerce *Coerce data as vistla would*

Description

This function will coerce the input vector into factor as vistla function would. Useful for testing or pre-computing quantisation.

Usage

```
vistla_coerce(x)
```

Arguments

x	Input vector.

Value

x coerced into factor.

write.dot *Export tree to a Graphviz DOT format*

Description

Exports the vistla tree in a DOT format, which can be later layouted and rendered by Graphiviz programs like dot or neato.

Usage

```
write.dot(
  x,
  con,
 vstyle = list(shape = function(x) ifelse(x$depth < 0, "egg", ifelse(x$leaf, "box",
    "ellipse")), label = function(x) sprintf("\"%s\"", x$name)),
 estyle = list(penwidth = function(x) sprintf("%0.3f", 0.5 + x$score/max(x$score) *
    2.5)),
  gstyle = list(overlap = "\"prism\"", splines = "true"),
  direction = c("none", "fromY", "intoY")
)
```

Arguments

x	vistla object.
con	connection; passed to writeLines. If missing, the DOT code is returned as a character vector.
vstyle	vertex attribute list — should be a named list of Graphviz attributes like shape or penwidth. For elements which are strings or numbers, the value is copied as is as an attribute value. For elements which functions, though, the function is called on a vistla_tree object and should return a vector of values.
estyle	edge attribute list, behaves exactly like vstyle. When functions are called, the Y-vertex is not present.
gstyle	graph attribute list. Functions are not supported here.
direction	when set to "none", graph is undirected, otherwise directed, for "fromY", root is a source, while for "intoY", a sink.

Value

For a missing con argument, a character vector with the graph in the DOT format, invisible NULL otherwise.

Note

Graphviz attribute values can be either strings, like "some vertex" in label, or atoms, like box for shape. When returning a string value, you must supply quotes, otherwise it will be included as an atom.

The default value of `gstyle` may invoke long layout calculations in Graphviz. Change to `list()` for a fast but less aesthetic layout.

The function does no validation whether provided attributes or values are correct.

Index