Package: dd4d (via r-universe)

December 18, 2024

Title Dummy Data for Dummies

Version 0.0.0.9000

Description Allows you to specify and sample from a Bayesian Network (a.k.a. a parametric Directed Acyclic Graph, or pDAG).

License MIT + file LICENSE

URL https://github.com/wjchulme/dd4d, https://opensafely-core.r-universe.dev/dd4d

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Imports rlang, dagitty, dplyr, purrr, tidyr, tibble, tidyselect, magrittr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/pak/sysreqs libicu-dev libssl-dev libnode-dev

Repository https://opensafely-core.r-universe.dev

RemoteUrl https://github.com/wjchulme/dd4d

RemoteRef HEAD

RemoteSha 0cca25dcf82cce0a60cd04572e55cc7c7ff8bb0e

Contents

`all_funs` *Get all functions that are used in a formula* `expr`.

Description

Get all functions that are used in a formula `expr`.

Usage

```
all_funs(expr)
```

Arguments

`expr`	a formula object

`bn2dagitty` *Converts a bn_df object to a dagitty object*

Description

Converts a bn_df object to a dagitty object

Usage

```
bn2dagitty(bn_df)
```

Arguments

`bn_df`	initialised bn_df object, with simulation instructions. Created with `bn_create`

Value

dagitty object

bn_create *Creates a bayesian network object from a list of nodes*

Description

Converts list to data frame which is a bit easier to work with, and embellishes with some useful columns. The function performs a few checks on the list, for instance to make sure the graph is acyclic and that variables used in the expressions are defined elsewhere or already known. The known_variables argument is for passing a character vector of variables names for variables that are already defined externally in a given dataset, which can be passed to bn_simulate whilst variable_formula is the variable name itself, this is to help with the bn_simulate function it doesn't actually lead to self-dependence (eg var depends on var)

Usage

```
bn_create(list, known_variables = NULL)
```

Arguments

list	of node objects, created by bn_node.
known_variables	character vector of variables that will be provided by an external dataset

Value

data.frame

bn_node *Specify a variable node in the network*

Description

Specify a variable node in the network

Usage

```
bn_node(variable_formula, missing_rate = ~0, keep = TRUE, needs = character())
```

Arguments

variable_formula	A RHS-only formula specified how to simulate that variable. Use ..n for the number of observations, which is later replaced by pop_size in the bn_simulate function.

missing_rate	A RHS-only formula. This specifies how missing values should be distributed. Can use a simple proportion such as ~0.5 or missingness can depend on other values for example using ~plogis(-2 + age*0.05), which says missingness increases with age.
keep	logical. Should this variable be kept in the final simulated output or not
needs	A character vector of variables. If any variables given in needs are missing / NA, then this variable is missing too.

Value

Object of class node and list.

Examples

```
bn_node(variable_formula = ~floor(rnorm(n=..n, mean=60, sd=15)))
```

---

bn_plot *Plot bn_df object*

Description

Plot bn_df object

Usage

```
bn_plot(bn_df, connected_only = FALSE)
```

Arguments

bn_df	initialised bn_df object, with simulation instructions. Created with bn_create
connected_only	logical. Only plot nodes that are connected to other nodes

Value

plot

bn_simulate *Simulate data from bn_df object*

Description

Simulate data from bn_df object

Usage

```
bn_simulate(bn_df, known_df = NULL, pop_size, keep_all = FALSE, .id = NULL)
```

Arguments

bn_df	initialised bn_df object, with simulation instructions. Created with bn_create
known_df	data.frame. Optional data.frame containing upstream variables used for simulation.
pop_size	integer. The size of the dataset to be created.
keep_all	logical. Keep all simulated variables or only keep those specified by keep
.id	character. Name of id column placed at the start of the dataset. If NULL (default) then no id column is created.

Value

tbl

rcat *Random categorical variables*

Description

Random categorical variables

Usage

```
rcat(n, levels, p)
```

Arguments

n	number of samples
levels	vector of categories to sample from
p	vector of probabilities

Value

a character vector

Examples

```
#' rcat(n=10, levels=c("a","b"), p=c(0.2,0.8))
```

`rfactor` *Random factor variables*

Description

Random factor variables

Usage

```
rfactor(n, levels, p)
```

Arguments

`n`	number of samples
`levels`	vector of categories to sample from
`p`	vector of probabilities

Value

a `factor` vector

Examples

```
#' rfactor(n=10, levels=c("a","b"), p=c(0.2,0.8))
```

`%ni%` *Inverse Value Matching*

Description

Complement of `%in%`. Returns the elements of `x` that are not in `y`.

Usage

```
x %ni% y
```

Arguments

`x`	a vector
`y`	a vector

Index