Package 'qmd'

October 13, 2022

Type Package

Title Quantification of Multivariate Dependence

Version 1.1.2

Description A multivariate copula-based dependence measure. For more information, see Griessenberger, Junker, Trutschnig (2022), On a multivariate copula-based dependence measure and its estimation, Electronic Journal of Statistics, 16, 2206-2251.

License GPL (>= 2)

Imports qad, Rcpp (>= 1.0.6), ggplot2, cowplot, dplyr, utils

Encoding UTF-8

RoxygenNote 7.1.2

LinkingTo Rcpp

NeedsCompilation yes

Author Nicolas Dietrich [aut, cre],
Florian Griessenberger [aut],
Robert R. Junker [aut],
Valentin Petztel [aut],
Wolfgang Trutschnig [aut]

Maintainer Nicolas Dietrich <nicolaspascal.dietrich@plus.ac.at>

Repository CRAN

Date/Publication 2022-08-22 09:20:05 UTC

R topics documented:

`.adaptive_masses` *Returns the sizes of the adaptive bins used for the adaptive ECBC for one vector.*

Description

Returns the sizes of the adaptive bins used for the adaptive ECBC for one vector.

Usage

```
.adaptive_masses(X, resolution)
```

Arguments

`X`	A vector, representing one sample of one variable
`resolution`	The resolution of the CB approximation

Value

A numeric vector of bin sizes

`.CB_make_cumulative_df` *Returns a list of reverse cumulative margins of a CB copula. The nth entry is thus the copula of X1,...,Xn*

Description

Returns a list of reverse cumulative margins of a CB copula. The nth entry is thus the copula of X1,...,Xn

Usage

```
.CB_make_cumulative_df(CB)
```

Arguments

`CB`	A matrix of CB weights.

Value

A list of CB weight matrixes of ascending dimension

`.EACBC` *Calculates an empirical CB approximation with adaptive bin sizes. This will be faster on data with many ties.*

Description

Calculates an empirical CB approximation with adaptive bin sizes. This will be faster on data with many ties.

Usage

```
.EACBC(X, resolution)
```

Arguments

`X`	A nxrho matrix of n samples of rho variables
`resolution`	The resolution of the CB approximation

Value

A matrix of dimension resolution^rho

`.EACBC_nonzero` *Returns non 0 entries of the EACBC*

Description

Returns non 0 entries of the EACBC

Usage

```
.EACBC_nonzero(X, resolution)
```

Arguments

`X`	A nxrho matrix of n samples of rho variables
`resolution`	The resolution of the CB approximation

Value

A list of local kernel masses

.ECBC *Calculates the empirical checkerboard approximation to some data.*

Description

Calculates the empirical checkerboard approximation to some data.

Usage

```
.ECBC(X, resolution)
```

Arguments

X	A nxrho matrix of n samples of rho variables
resolution	The resolution of the CB approximation

Value

A matrix of dimension resolution^rho

.local_kernel_integral *Computes the D1-difference of two CB matrizes on a local CB dimension*

Description

Computes the D1-difference of two CB matrizes on a local CB dimension

Usage

```
.local_kernel_integral(k1, k2, y)
```

Arguments

k1	Vector of local CB weights of first matrix
k2	Vector of local CB weights of second matrix
y	Vector indicating the bin sizes of the local dimension

Value

number indicating the difference between k1 and k2

.random_CB *Creates a random CB copula of resolution 2^steps*

Description

Creates a random CB copula of resolution 2^steps

Usage

```
.random_CB(rho, steps, de, ie)
```

Arguments

rho	The number of variables
steps	Number of iteration steps, the final resolution will be 2^steps
de	Exponent to increase dependence
ie	Exponent to increase independence

Value

A matrix of dimension (2^steps)^rho

.sample_CB *Generate a sample of some CB copula-*

Description

Generate a sample of some CB copula-

Usage

```
.sample_CB(CB, n)
```

Arguments

CB	A weight matrix of a CB copula
n	The number of samples to be generated

Value

Matrix of dimension nxm where m is the dimension of CB

ECBC *Compute empirical checkerboard copula in arbitrary dimension*

Description

The function ECBC computes the mass distribution of the empirical (checkerboard) copula, given a rho-dimensional sample X. If resolution equals sample size, the bi-linearly extended empirical copula is returned. Note, if there are ties in the sample an adjusted empirical copula is calculated. If bin.size is set to "adaptive" the sizes of the bins will be adjusted to fit the data without overspilling into neighboring bins. This might affects the result, but is more efficient with samples having many ties as no adjustment is needed.

Usage

```
ECBC(X, resolution, bin.size = "fixed")
```

Arguments

X	a numeric matrix of dimension rho indicating a sample of rho variables
resolution	an integer indicating the resolution N of the checkerboard copula
bin.size	either "fixed" or "adaptive", indicating whether the checkerboard copula may vary its bin sizes (defaults to "fixed")

Value

array of dimension resolution^rho.

Examples

```
n <- 1000
x1 <- runif(n)
x2 <- runif(n)
y <- x1 + x2 + rnorm(n)
M <- ECBC(X = cbind(x1,x2,y), resolution = 8)
```

feature_selection *Variable selection using the qmd-dependence values*

Description

Given a d-dimensional random vector X containing the explanatory variables and a uni-variate response variable y, this function uses the qmd-dependence values to select the most relevant (influential) explanatory variables. Two different methods are available and are explained in the section Details.

Usage

```
feature_selection(
  X,
  y,
  method = "combVar",
  bin.size = "fixed",
  plot = TRUE,
  na.exclude = FALSE,
  max_num_features = NULL,
  plot.title = NULL,
  plot.color = "hotpink"
)
```

Arguments

X	a numeric matrix or data.frame of dimension d containing the explanatory variables
y	a numeric vector containing the uni-variate response variable
method	possible options are c("combVar", "addVar"), see Details.
bin.size	either "fixed", "adaptive" or "sparse.adaptive", indicating whether the checkerboard copula may vary its bin sizes (defaults to "fixed"). Setting this to "adaptive" might affect the results but will be faster if the sample has many ties.
plot	logical indicating whether the feature selection plot is printed
na.exclude	logical if all rows containing NAs should be removed.
max_num_features	maximal number of explanatory variables to be selected
plot.title	a label for the title
plot.color	a colour for the selected variables

Details

method 1 (default) - "combVar": computes all qmd-dependence scores, i.e., calculates the dependence of every combination of explanatory variables to the response variable y and selects for each number of explanatory variables the combination with the greatest dependence score. This procedure is computational expensive and is only available up to 15 explanatory variables.

method 2 - "addVar": stepwise procedure which calculates all bi-variate dependence values q(X_i,Y) and selects the variable X_j exhibiting the greatest dependence value. In the next step all three-dimensional combinations q((X_j, X_i), Y) (for every i =1,..., d and i not j) are computed and the variable exhibiting again the greatest dependence score is added. In this manner the procedure works up to dimension d.

Value

a list containing a data.frame (result) and the corresponding plots. The data.frame result contains the number of explanatory variables (nVars), the combination of selected variables (selVars), the dependence measure zeta1 (qmd) of the selected variables to the response y and the resolution of

the empirical checkerboard copula (ECBC_resolution). For the method "combVar" the dependence value zeta1 (qmd) is returned for all combinations of explanatory variables and is sorted in decreasing order according to zeta1.

Examples

```
n <- 1000
x1 <- runif(n)
x2 <- rexp(n)
x3 <- x1 + log(x2) + rnorm(n)
x4 <- rnorm(n)
x5 <- x4^2
x6 <- x1 + x5 + rnorm(n)
x7 <- 1:n
y <- x2 + x4*x7 + runif(n)
X <- data.frame(x1,x2,x3,x4,x5,x6,x7)
fit <- feature_selection(X, y, method = "combVar", plot = TRUE)
fit <- feature_selection(X, y, method = "addVar", plot = TRUE)
```

---

qmd *Quantification of Multivariate Dependence*

Description

Function for estimating the non-parametric copula-based multivariate measure of dependence $\zeta 1$. This measure quantifies the extent of dependence between a d-dimensional random vector X and a uni-variate random variable y (i.e., it measures the influence of d explanatory variables X1,...,Xd on a univariate variable y). Further details can be found in the section Details and the corresponding references.

Usage

```
qmd(
  X,
  y,
  ties.correction = FALSE,
  resolution = NULL,
  p.value = FALSE,
  R = 1000,
  print = TRUE,
  na.exclude = FALSE
)
```

Arguments

X	a numeric matrix or data.frame of dimension d containing the explanatory variables
y	a numeric vector containing the uni-variate response variable

`ties.correction`
: logical indicating if the measure of dependence should be calculated with ties-correction (experimental version). Default = FALSE.

`resolution`
: an integer indicating the resolution N of the checkerboard aggregation. We recommend to use the default configuration (resolution = NULL), which uses the resolution N(n) = floor(n^(1/(d+1))), where d denotes the number of explanatory variables.

`p.value`
: logical indicating if a p-value is returned using permutations of Y

`R`
: integer indicating the number of repetitions for the calculation of the p-value (default = 1000)

`print`
: logical indicating whether the results of the function are printed

`na.exclude`
: logical if all rows containing NAs should be removed.

Details

In the following we will simply write q for the dependence measure $\zeta 1$. Furthermore, X denotes a random vector consisting of d random variables and y denotes a univariate random variable. Then the theoretical dependence measure q fulfills the following essential properties of a dependence measure:

- [N] q(X,y) attains values in [0,1] (normalization).
- [I] q(X,y) = 0 if and only if X and y are independent (independence).
- [C] q(X,y) = 1 if and only if y is a function of X (complete dependence).

Further properties of q and the exact mathematical definition can be found in Griessenberger et al. (2022). This function qmd() contains the empirical checkerboard-estimator (ECB-estimator), which is strongly consistent and attains always positive values between 0 and 1. Note, that interpretation of low values has to be done with care and always under consideration of the sample size. For instance, values of 0.2 can point towards independence in small sample settings. An additional p-value (testing for independence and being based on permutations of y) helps in order to correctly understand the dependence values. Since independence constitutes the null hypothesis a p-value above the significance level (e.g., 0.05) indicates independence between X and y.

Value

qmd returns a list object containing the following components:

- input: data containing the explanatory variables (X)
- output: data containing the response (y)
- q(X,y): dependence measure indicating the extent of dependence between X and y
- results: data.frame containing the dependence measure and the corresponding p-value
- resolution: an integer indicating the resolution of the aggregated checkerboard copula
- Sample size

References

Griessenberger, F., Junker, R.R. and Trutschnig, W. (2022). On a multivariate copula-based dependence measure and its estimation, Electronic Journal of Statistics, 16, 2206-2251.

Examples

```
#(complete dependence for dimension 4)
n <- 300
x1 <- runif(n)
x2 <- runif(n)
x3 <- x1 + x2 + rnorm(n)
y <- x1 + x2 + x3
qmd(X = cbind(x1,x2,x3), y = y, p.value = TRUE)

#(independence for dimension 4)
n <- 500
x1 <- runif(n)
x2 <- runif(n)
x3 <- x1 + x2 + rnorm(n)
y <- runif(n)
qmd(X = cbind(x1,x2,x3), y = y, p.value = TRUE)

#(binary output (classification) for dimension 3)
n <- 500
x1 <- runif(n)
x2 <- runif(n)
y <- ifelse(x1 + x2 < 1, 0, 1)
qmd(X = cbind(x1,x2), y = y, p.value = TRUE)
#(independence)
y <- runif(n)
qmd(X = cbind(x1,x2), y = y, p.value = TRUE)
```

---

qmdrank *Equivalent to rank(x, ties.method = "max") but not as stupidly slow*

Description

Equivalent to rank(x, ties.method = "max") but not as stupidly slow

Usage

```
qmdrank(x)
```

Arguments

x	A numeric vector

Value

An integer vector specifying for each value in x the rank within x. If one value appears multiple time the maximum is used.

seq_until_changes *Returns a vector*

Description

Returns a vector

Usage

```
seq_until_changes(x)
```

Arguments

x	A usually sorted vector

Value

A sequence along x. If consecutive values in x are equal the maximal value is used.

zeta1 *Multivariate dependence measure*

Description

Function for estimating the non-parametric copula-based multivariate measure of dependence $\zeta 1$. This measure quantifies the extent of dependence between a d-dimensional random vector X and a uni-variate random variable y (i.e., it measures the influence of d explanatory variables X1,...,Xd on a univariate variable y).

Usage

```
zeta1(X, y, ties.correction = FALSE, bin.size = "fixed", resolution = NULL)
```

Arguments

X	a numeric matrix or data.frame of dimension d containing the explanatory variables
y	a numeric vector containing the uni-variate response variable
ties.correction	logical indicating if the measure of dependence should be calculated with ties-correction (experimental version). Default = FALSE.
bin.size	either "fixed", "adaptive" or "sparse.adaptive", indicating whether the checkerboard copula may vary its bin sizes (defaults to "fixed"). Setting this to "adaptive" might affect the results but will be faster if the sample has many ties.

`resolution` an integer indicating the resolution N of the checkerboard aggregation. We recommend to use the default configuration (resolution = NULL), which uses the resolution N(n) = floor(n^(1/(d+1))), where d denotes the number of explanatory variables.

Details

see function qmd(...).

Value

A numeric value indicating the extent of dependence between the vector X and the variable y (or, equivalently, the influence of X on y).

References

Griessenberger, F., Junker, R.R. and Trutschnig, W. (2022). On a multivariate copula-based dependence measure and its estimation, Electronic Journal of Statistics, 16, 2206-2251.

Examples

```
#(complete dependence for dimension 4)
n <- 300
x1 <- runif(n)
x2 <- runif(n)
x3 <- x1 + x2 + rnorm(n)
y <- x1 + x2 + x3
zeta1(X = cbind(x1,x2,x3), y = y)

#(independence for dimension 4)
n <- 500
x1 <- runif(n)
x2 <- runif(n)
x3 <- x1 + x2 + rnorm(n)
y <- runif(n)
zeta1(X = cbind(x1,x2,x3), y = y)

#(binary output for dimension 3)
n <- 500
x1 <- runif(n)
x2 <- runif(n)
y <- ifelse(x1 + x2 < 1, 0, 1)
zeta1(X = cbind(x1,x2), y = y)
```

Index