Package 'CPC'

April 8, 2024

Title Implementation of Cluster-Polarization Coefficient

Version 2.6.0

Description Implements cluster-polarization coefficient for measuring distributional polarization in single or multiple dimensions, as well as associated functions. Contains support for hierarchical clustering, k-means, partitioning around medoids, density-based spatial clustering with noise, and manually imposed cluster membership. Mehlhaff (forthcoming) <doi:10.1017/S0003055423001041>.

License CC0

Encoding UTF-8

RoxygenNote 7.2.0

URL `https://imehlhaff.net/CPC/`

BugReports `https://github.com/imehlhaff/CPC/issues`

Imports stats, cluster, dbscan, Rfast

NeedsCompilation no

Author Isaac Mehlhaff [aut, cre, cph] (<`https://orcid.org/0000-0001-5776-005X`>)

Maintainer Isaac Mehlhaff <`isaac.mehlhaff@gmail.com`>

Repository CRAN

Date/Publication 2024-04-08 16:20:06 UTC

R topics documented:

correlate *Test for Bivariate Correlation*

Description

Calculates correlation coefficient between two variables and returns a list containing the correlation estimate, its standard error, the p-value of a null-hypothesis significance test, and the number of observations used.

Usage

```
correlate(x, y, ...)
```

Arguments

`x`	a numeric vector.
`y`	a numeric vector.
`...`	arguments passed to `cor.test()`.

Details

Additional arguments to alter the type of null hypothesis significance test, the method used to calculate the correlation coefficient, the confidence level, or other options should be passed to `correlate()` and will be inherited by `cor.test()`. Note that unlike `cor.test()`, both arguments `x` and `y` are required.

Value

Returns a list with elements containing the correlation coefficient estimate, its associated standard error, the p-value of a null-hypothesis significance test, and the number of observations used, all as numeric vectors of length 1.

Examples

```
data <- matrix(c(rnorm(50, 0, 1), rnorm(50, 5, 1)), ncol = 2, byrow = TRUE)

correlate(data[, 1], data[, 2])
```

`CPC` *Cluster-Polarization Coefficient*

Description

Implements clustering algorithms and calculates cluster-polarization coefficient. Contains support for hierarchical clustering, k-means clustering, partitioning around medoids, density-based spatial clustering with noise, and manual assignment of cluster membership.

Usage

```
CPC(
  data,
  type,
  k = NULL,
  epsilon = NULL,
  model = FALSE,
  adjust = FALSE,
  cols = NULL,
  clusters = NULL,
  ...
)
```

Arguments

`data` a numeric vector or `n x k` matrix or data frame. If `type` = `"manual"`, `data` must be a matrix containing a vector identifying cluster membership for each observation, to be passed to `clusters` argument.

`type` a character string giving the type of clustering method to be used. See Details.

`k` the desired number of clusters. Required if `type` is one of `"hclust"`, `"diana"`, `"kmeans"`, or `"pam"`.

`epsilon` radius of epsilon neighborhood. Required if `type` = `"dbscan"`.

`model` a logical indicating whether clustering model output should be returned. Defaults to `FALSE`.

`adjust` a logical indicating whether the adjusted CPC should be calculated. Defaults to `FALSE`. Note that both CPC and adjusted CPC are automatically calculated and returned if `model` = `TRUE`.

`cols` columns of `data` to be used in CPC calculation. Only used if `type` = `"manual"`.

`clusters` column of `data` indicating cluster membership for each observation. Only used if `type` = `"manual"`.

`...` arguments passed to other functions.

Details

type must take one of six values:
"hclust": agglomerative hierarchical clustering with hclust(),
"diana": divisive hierarchical clustering with diana(),
"kmeans": k-means clustering with kmeans(),
"pam": k-medoids clustering with pam(),
"dbscan": density-based clustering with dbscan(),
"manual": no clustering is necessary, researcher has specified cluster assignments.

For all clustering methods, additional arguments to fine-tune clustering performance, such as the specific algorithm to be used, should be passed to CPC() and will be inherited by the specified clustering function. In particular, if type = "kmeans", using a large number of random starts is recommended. This can be specified with the nstart argument to kmeans(), passed directly to CPC().

If type = "manual", data must contain a vector identifying cluster membership for each observation, and cols and clusters must be defined.

Value

If model = TRUE, CPC() returns a list with components containing output from the specified clustering function, all sums of squares, the CPC, the adjusted CPC, and associated standard errors. If model = FALSE, CPC() returns a numeric vector of length 1 giving the CPC (if adjust = FALSE) or adjusted CPC (if adjust = TRUE).

Examples

```
data <- matrix(c(rnorm(50, 0, 1), rnorm(50, 5, 1)), ncol = 2, byrow = TRUE)
clusters <- matrix(c(rep(1, 25), rep(2, 25)), ncol = 1)
data <- cbind(data, clusters)

CPC(data[,c(1:2)], "kmeans", k = 2)
CPC(data, "manual", cols = 1:2, clusters = 3)
```

CPCdata.frame *Data Manipulation for CPC Calculation*

Description

Converts numeric matrix to data frame with necessary format for "manual" CPC() calculation.

Usage

```
CPCdata.frame(data, cols, clusters)
```

Arguments

`data`	a numeric `n x k` matrix or data frame.
`cols`	columns in `data` to be used for calculating `CPC()`.
`clusters`	column in `data` giving cluster membership.

Value

Returns a data frame with dimensions identical to those of `data`.

Examples

```
data <- matrix(c(rnorm(50, 0, 1), rnorm(50, 5, 1)), ncol = 2, byrow = TRUE)
clusters <- matrix(c(rep(1, 25), rep(2, 25)), ncol = 1)
data <- cbind(data, clusters)
CPCdata.frame(data, 1:2, 3)
```

`diff_multidim` *Multidimensional Difference-in-Means*

Description

Calculates average Euclidean distance between means in arbitrary dimensions.

Usage

```
diff_multidim(data, cols, clusters)
```

Arguments

`data`	a numeric vector or `n x k` matrix or data frame containing a vector identifying cluster membership for each observation, to be passed to `clusters` argument.
`cols`	columns of `data` to be used in difference-in-means calculation.
`clusters`	column of `data` indicating cluster membership for each observation.

Value

Returns a numeric vector of length 1.

Examples

```
data <- matrix(c(rnorm(50, 0, 1), rnorm(50, 5, 1)), ncol = 2, byrow = TRUE)
clusters <- matrix(c(rep(1, 25), rep(2, 25)), ncol = 1)
data <- cbind(data, clusters)

diff_multidim(data, 1:2, 3)
```

`Euclidean` *Euclidean Distance from Dimension Means*

Description

Calculates two-dimensional Euclidean distance between all points and dimension means.

Usage

```
Euclidean(data)
```

Arguments

`data`	an `n x 2` matrix or data frame.

Value

Returns a numeric vector of length 1.

Examples

```
data <- matrix(c(rnorm(50, 0, 1), rnorm(50, 5, 1)), ncol = 2, byrow = TRUE)

Euclidean(data)
```

`SS` *Sum-of-Squares Calculation*

Description

Calculates sums of squares for uni- or multi-dimensional numeric data using the distance matrix.

Usage

```
SS(data, ...)
```

Arguments

`data`	a numeric vector or `n x k` matrix or data frame.
`...`	arguments passed to `dist()`.

Value

Returns a numeric vector of length 1.

Examples

```
data <- matrix(c(rnorm(50, 0, 1), rnorm(50, 5, 1)), ncol = 2, byrow = TRUE)
SS(data)
```

Index