Package ‘ClusterSignificance’

October 12, 2016

Title Investigates Significance of Clusters by Reducing the Data to One Dimension to be Able to Easy Set a Score for the Separation, and a p-Value is then Calculated from Permutations of the Original Data

Version 1.0.3

Author Jason T. Serviss and Jesper R. Gadin

Maintainer Jason T Serviss `<jason.serviss@ki.se>`

Description The ClusterSignificance package provides tools to assess if clusters have a separation different from random or permuted data. ClusterSignificance investigates clusters of two or more groups by first, projecting all points onto a one dimensional line. Cluster separations are then scored and the probability of the seen separation being due to chance is evaluated using a permutation method.

Depends R (>= 3.3.0)

Imports methods, pracma, princurve, scatterplot3d, RColorBrewer, grDevices, graphics, utils

License GPL-3

LazyData true

Suggests knitr, rmarkdown, testthat, BiocStyle, ggplot2, plsgenomics

VignetteBuilder knitr

biocViews Clustering, Classification, PrincipalComponent, StatisticalMethod

NeedsCompilation no

Collate 'ClusterSignificance-package.R' 'All-classes.R' 'classifier-methods.R' 'initialize-methods.R' 'mlpMatrix.R' 'pcpMatrix.R' 'permutation-methods.R' 'plot-methods.R' 'projection-methods.R' 'show-methods.R'

RoxygenNote 5.0.1

R topics documented:

`ClusterSignificance-package`

A package with purpose of giving a p-value to separations of clusters.

Description

It is often a common wish after clustering procedure to assess how significant a seen separation is. This package investigates clusters of two or more groups by projecting all points to a line or curve. The placement on this line is used for classification of the points and the probability of the seen separation just being because of chance is evaluated using a permutation method.

Details

Package:	ClusterSignificance
Type:	Package
Version:	1.0
Date:	2016-02-28
License:	GPL-3

Author(s)

Author: Jason T. Serviss, Jesper R. Gadin

References

Reference to published application note (work in progress)

`ClassifiedPoints-class` *classification of the one dimensional points in a Pcp or Mlp object*

Description

classification based on ROC params (TN TP FP FN)

Usage

```
## S4 method for signature 'ClassifiedPoints'
getData(x, n = NULL)

classify(x, ...)

## S4 method for signature 'Pcp'
classify(x, ...)

## S4 method for signature 'Mlp'
classify(x, ...)

## S4 method for signature 'ClassifiedPoints'
initialize(.Object, ..., scores,
  scores.points = scores.points, scores.index = scores.index, ROC)

## S4 method for signature 'ClassifiedPoints,missing'
plot(x, y, ...)

## S4 method for signature 'ClassifiedPoints'
show(object)
```

Arguments

`x`	Pcp or Mlp Object for the function classify otherwise it is a ClassifiedPoints object
`n`	data to extract from ClassifiedPoints (NULL gives all)
`...`	additional arguments to pass on
`.Object`	internal object
`scores`	final scores
`scores.points`	sorted points
`scores.index`	index of sorted points
`ROC`	parameters (TN, TP, FN and FP)
`y`	default plot param, which should be set to NULL
`object`	ClassifiedPoints Object

Details

Tests all possible discrimination lines and picks the one with highest score based on a score which is simply calculated by formula (TP - FP) + (TN - FN)

The plot shows the distribution of scores for different discrimination lines Each line is a separator that creates a score for the separation of two groups, and the height of the line marks the score for this separation.

Value

The classify function returns an object of class ClassifiedPoints

Author(s)

Jesper R. Gadin and Jason T. Serviss

Examples

```
#use demo data
data(pcpMatrix)
groups <- rownames(pcpMatrix)

#run function
prj <- pcp(pcpMatrix, groups)
cl <- classify(prj)

#getData accessor
getData(cl)

#getData accessor specific
getData(cl, "scores")

#plot result
plot(cl)
```

`Mlp-class` *projection of points into one dimension*

Description

project points onto the mean based line

Usage

```
## S4 method for signature 'Mlp'
getData(x, n = NULL)

## S4 method for signature 'Mlp'
initialize(.Object, ..., groups, points.orig, line,
  points.onedim)

## S4 method for signature 'Mlp,missing'
plot(x, y, steps = "all", ...)

mlp(mat, ...)

## S4 method for signature 'matrix'
mlp(mat, groups, ...)

## S4 method for signature 'Mlp'
show(object)
```

Arguments

`x`	matrix object for the function mlp otherwise it is a Mlp object
`n`	data to extract from Mlp (NULL gives all)
`.Object`	internal object
`...`	additional arguments to pass on
`groups`	vector in same order as rows in matrix
`points.orig`	multidimensional points describing the original data
`line`	multidimensional points describing a line
`points.onedim`	a vector of points
`y`	default plot param, which should be set to NULL(default: NULL)
`steps`	1,2,3,4,5,6 or "all"
`mat`	matrix with samples on rows, PCs in columns. Ordered PCs, with PC1 to the left.
`object`	Mlp object

Details

projection of the points on a line separating the mean of two groups. mlp is the abbreviation of 'mean line projection'. The Function accepts at the moment only two groups and two PCs at the time.

An Object containing results from a mean line projection reduction to one dimension.

The group and the one dimensional points are the most important information to carry out a classification using the classify() function. But as a help to illustrate the details of the dimension reduction, the information from some critical steps are stored in the object. To visually explore these there is a dedicated plot method for Mlp objects, use plot().

Value

The mlp function returns an object of class Mlp

Author(s)

Jesper R. Gadin and Jason T. Serviss

Examples

```
#use demo data
data(mlpMatrix)
groups <- rownames(mlpMatrix)

#run function
prj <- mlp(mlpMatrix, groups)

#getData accessor
getData(prj)

#getData accessor specific
getData(prj, "line")

#plot result
plot(prj)
```

`mlpMatrix` *Simulated data used to demonstrate the Mlp method.*

Description

Mlp demonstration matrix.

Usage

```
mlpMatrix
```

Format

Matrix

rownames Groups

colnames dimension number

Value

simulated matrix

Examples

```
mlpMatrix
```

Pcp-class *projection of points into one dimension*

Description

project points onto a principal curve

Usage

```
getData(x, ...)

## S4 method for signature 'Pcp'
getData(x, n = NULL)

## S4 method for signature 'Pcp'
initialize(.Object, ..., groups, points.orig, line,
  points.onedim, index)

## S4 method for signature 'Pcp,missing'
plot(x, y, steps = "all", ...)

pcp(mat, ...)

## S4 method for signature 'matrix'
pcp(mat, groups, ...)

## S4 method for signature 'Pcp'
show(object)
```

Arguments

`x`	matrix object for the function pcp otherwise it is a Pcp object
`...`	additional arguments to pass on
`n`	data to extract from Pcp (NULL gives all)
`.Object`	internal object
`groups`	vector in same order as rows in matrix
`points.orig`	multidimensional points describing the original data
`line`	multidimensional points describing a line
`points.onedim`	a vector of points
`index`	internal index from the projection

`y`	default plot param, which should be set to NULL
`steps`	1,2,3,4,5,6 or "all"
`mat`	matrix with samples on rows, PCs in columns. Ordered PCs, with PC1 to the left.
`object`	Pcp object

Details

The resulting Pcp object containing results from a principal curve reduction to one dimension. The group and the one dimensional points will be the information needed to carry out a classification using the classify() function. But as a help to illustrate the details of the dimension reduction , the information from some critical steps are stored in the object. To visually explore these there is a dedicated plot method for Php objects, use plot().

Value

The pcp function returns an object of class Pcp

Author(s)

Jesper R. Gadin and Jason T. Serviss

Examples

```
#use demo data
data(pcpMatrix)
groups <- rownames(pcpMatrix)

#run function
prj <- pcp(pcpMatrix, groups)

#getData accessor
getData(prj)

#getData accessor specific
getData(prj, "line")

#plot the result (if dim >2, then plot in 3d)
plot(prj)

#plot the result (if dim =2, then plot in 2d)
prj2 <- pcp(pcpMatrix[,1:2], groups)
plot(prj2)
```

`pcpMatrix` *Simulated data used to demonstrate the Pcp method.*

Description

Pcp demonstration matrix.

Usage

```
pcpMatrix
```

Format

Matrix

rownames Groups

colnames dimension number

Value

simulated matrix

Examples

```
pcpMatrix
```

`PermutationResults-class`
permutation test

Description

Test how the classification performs compared to random (eg. permuted) data.

Usage

```
## S4 method for signature 'PermutationResults'
getData(x, n = NULL)

## S4 method for signature 'PermutationResults'
c(x, ..., recursive = FALSE)

pvalue(x, ...)

## S4 method for signature 'PermutationResults'
```

```
pvalue(x, ...)

## S4 method for signature 'PermutationResults'
initialize(.Object, ..., scores.real, scores.vec)

permute(mat, ...)

## S4 method for signature 'matrix'
permute(mat, groups, projmethod = "pcp", iter = 100,
  user.permutations = NULL, seed = 3, verbose = TRUE, ...)

## S4 method for signature 'PermutationResults,missing'
plot(x, y, ...)

## S4 method for signature 'PermutationResults'
show(object)
```

Arguments

`x`	matrix for the function permute, otherwise it is a PermutationResults object
`n`	data to extract from ClassifiedPoints (NULL gives all)
`...`	arguments to pass on
`recursive`	dont use (belongs to default generic of combine 'c()')
`.Object`	internal object
`scores.real`	the real score
`scores.vec`	all permuted scores
`mat`	matrix with samples on rows, PCs in columns. Ordered PCs, with PC1 to the left.
`groups`	vector in same order as rows in matrix
`projmethod`	'pcp' or 'mlp'
`iter`	integer number of iterations to be performed.
`user.permutations`	user defined permutation matrix
`seed`	random seed to be used by the internal permutation
`verbose`	makes function more talkative
`y`	default plot param, which should be set to NULL
`object`	ClassifiedPoints Object

Details

This is a test suit and will return a summarized object. The default of the parameter 'iter' is set quite low, and in principle the more iterations the better, or until the pvalue converges to a specifc value. If no pre-permuted data has been supplied by the user, then the internal permutation method will perform a sampling without replacement within each dimension.

Value

The permute function returns an object of class PermutationResults

Author(s)

Jesper R. Gadin and Jason T. Serviss

Examples

```
#use pcp method
data(pcpMatrix)
groups <- rownames(pcpMatrix)

#run function
iterations <- 10
pe <- permute(
    mat=pcpMatrix,
    groups=groups,
    iter=iterations,
    projmethod="pcp"
)

#use mlp method
data(mlpMatrix)
groups <- rownames(mlpMatrix)
pe <- permute(
    mat=mlpMatrix,
    groups=groups,
    iter=iterations,
    projmethod="mlp"
)


#getData accessor
getData(pe)

#getData accessor specific
getData(pe, "scores.vec")

#get pvalue
pvalue(pe)

#plot result
plot(pe)

#combine three (parallell) jobs on the same matrix
pe2 <- c(pe, pe, pe)
```

Index