Package ‘LearnClust’

November 29, 2020

Type Package
Date 2020-11-24
Title Learning Hierarchical Clustering Algorithms
Version 1.1
Author Roberto Alcantara [aut, cre],
Juan Jose Cuadrado [aut],
Universidad de Alcala de Henares [aut]
Maintainer Roberto Alcantara <roberto.alcantara@edu.uah.es>
Description Classical hierarchical clustering algorithms, agglomerative and divisive clustering. Algorithms are implemented as a theoretical way, step by step.
   It includes some detailed functions that explain each step. Every function allows options to get different results using different techniques.
   The package explains non expert users how hierarchical clustering algorithms work.
License Unlimited
Depends magick
Suggests knitr, rmarkdown
VignetteBuilder knitr
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
NeedsCompilation no
Repository CRAN
Date/Publication 2020-11-29 22:50:02 UTC

R topics documented:

agglomerativeHC .................................................. 3
agglomerativeHC.details ...................................... 4
canberradistance ............................................... 5
canberradistance.details ................................. 6
R topics documented:

canberradistanceW .......................... 7
canberradistanceW.details ........................ 8
chebyshevDistance .......................... 9
chebyshevDistance.details ........................ 10
chebyshevDistanceW .......................... 11
chebyshevDistanceW.details ........................ 12
clusterDistance ............................ 13
clusterDistance.details ........................ 14
clusterDistanceByApproach ........................ 15
clusterDistanceByApproach.details ........................ 16
complementaryClusters ........................ 17
complementaryClusters.details ........................ 18
correlationHC .............................. 19
correlationHC.details ........................ 21
distances ................................. 22
distances.details .............................. 23
divisiveHC ................................. 24
divisiveHC.details ............................ 26
distance ................................. 27
distance.details .............................. 28
distanceW ................................. 29
distanceW.details ............................ 30
getCluster ................................. 31
getCluster.details ............................ 32
getClusterDivisive ........................... 33
getClusterDivisive.details ........................ 34
initClusters ............................... 35
initClusters.details ........................... 36
initData ............................... 37
initData.details ............................. 38
initImages ............................... 39
initTarget ............................... 39
initTarget.details ............................. 40
matrixDistance ............................ 41
maxDistance .............................. 42
maxDistance.details ........................... 43
mdAgglomerative ............................ 44
mdAgglomerative.details ........................ 45
mdDivisive ............................... 46
mdDivisive.details ............................ 47
mdistance ............................... 48
mdistance.details ............................ 49
mdistanceW ............................... 50
mdistanceW.details ............................ 51
minDistance .............................. 52
minDistance.details ........................... 53
newCluster ............................... 54
newCluster.details ............................ 55
To execute agglomerative hierarchical clusterization algorithm by distance and approach.

Usage

agglomerativeHC(data, distance, approach)

Arguments

data could be a numeric vector, a matrix or a numeric data frame. It will be transformed into matrix and list to be used.
distance is a string. It chooses the distance to use.
approach is a string. It chooses the approach to use.

Details

This function is the main part of the agglomerative hierarchical clusterization method. It executes the theoretical algorithm step by step.

1 - The function transforms data in useful object to be used.
2 - It creates the clusters.
3 - It calculates a matrix distances with the clusters created applying distance and approach given.
4 - It chooses the distance value and gets the clusters.
5 - It groups the clusters in a new one and updates clusters list.
6 - It repeats these steps until an unique cluster exists.
agglomerativeHC.details

Value

R object with a dendrogram, the grouped clusters and the list with every cluster.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
a <- c(1,2,1,3,1,4,1,5,1,6)
matrixA <- matrix(a,ncol=2)
dataFrameA <- data.frame(matrixA)
agglomerativeHC(a,'EUC','MAX')
agglomerativeHC(matrixA,'MAN','AVG')
agglomerativeHC(dataFrameA,'CAN','MIN')
```

Description

To explain agglomerative hierarchical clusterization algorithm by distance and approach.

Usage

```r
agglomerativeHC.details(data, distance, approach)
```

Arguments

data could be a numeric vector, a matrix or a numeric data frame. It will be transformed into matrix and list to be used.
distance is a string. It chooses the distance to use.
approach is a string. It chooses the approach to use.
canberradistance

Details

This function is the main part of the agglomerative hierarchical clusterization method. It explains the theoretical algorithm step by step.

1 - The function transforms data into a useful object to be used.
2 - It creates the clusters.
3 - It calculates a matrix distance with the clusters created by applying the distance and the approach given.
4 - It chooses the distance value and gets the clusters.
5 - It groups the clusters in a new one and updates the clusters list.
6 - It repeats these steps until an unique cluster exists.

Value

agglomerative algorithm explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
a <- c(1,2,1,3,1,4,1,5,1,6)
matrixA <- matrix(a,ncol=2)
dataFrameA <- data.frame(matrixA)
agglomerativeHC.details(a,'EUC','MAX')
agglomerativeHC.details(matrixA,'MAN','AVG')
agglomerativeHC.details(dataFrameA,'CAN','MIN')
```

---

canberradistance  | To calculate the Canberra distance.

Description

To calculate the Canberra distance of two clusters.
Usage

canberradistance(x, y)

Arguments

x is a numeric vector or a matrix. It represents the values of a cluster.
y is a numeric vector or a matrix. It represents the values of a cluster.

Details

This function is part of the hierarchical clusterization method. The function calculates the Canberra distance value from \( x \) and \( y \).

Value

canberra distance value.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

\[
\begin{align*}
  x & \leftarrow c(1, 2) \\
  y & \leftarrow c(1, 3) \\
  \text{cluster1} & \leftarrow \text{matrix}(x, \text{ncol}=2) \\
  \text{cluster2} & \leftarrow \text{matrix}(y, \text{ncol}=2) \\
  \text{canberradistance}(x, y) \\
  \text{canberradistance}(& \text{cluster1, cluster2})
\end{align*}
\]

Description

To show the formula and to return the Canberra distance of two clusters.

Usage

canberradistance.details(x, y)
Arguments

x is a numeric vector or a matrix. It represents the values of a cluster.

y is a numeric vector or a matrix. It represents the values of a cluster.

Details

This function is part of the hierarchical clusterization method. The function calculates the Canberra distance value from x and y.

Value

canberra distance value with its formula.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
x <- c(1,2)
y <- c(1,3)
cluster1 <- matrix(x, ncol=2)
cluster2 <- matrix(y, ncol=2)
canberradistance(x, y)
canberradistance(cluster1, cluster2)
```

---

`canberradistanceW`  
To calculate the Canberra distance applying weights.

Description

To calculate the Canberra distance between clusters applying weights given.

Usage

canberradistanceW(cluster1, cluster2, weight)
Arguments

cluster1 is a cluster.
cluster2 is a cluster.
weight is a numeric vector.

Details

The function calculates the Canberra distance value from cluster1 and cluster2, applying weights to the cluster's components.

Value

canberra distance applying weights value.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)
weight1 <- c(0.4,0.6)
weight2 <- c(2,12)

canberradistanceW(cluster1,cluster2,weight1)
canberradistanceW(cluster1,cluster2,weight2)

canberradistanceW.details

To calculate the Canberra distance applying weights.

Description

To explain how to calculate the Canberra distance between clusters applying weights given.

Usage

canberradistanceW.details(cluster1, cluster2, weight)
chebyshevDistance

Arguments

cluster1 is a cluster.
cluster2 is a cluster.
weight is a numeric vector.

Details

The function calculates the Canberra distance value from cluster1 and cluster2, applying weights to the cluster’s components.

Value

canberra distance applying weights value. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)
weight1 <- c(0.4,0.6)
weight2 <- c(2,12)
canberradistanceW.details(cluster1,cluster2,weight1)
canberradistanceW.details(cluster1,cluster2,weight2)

chebyshevDistance

To calculate the Chebyshev distance.

Description

To calculate the Chebyshev distance of two clusters.

Usage

chebyshevDistance(x, y)
Arguments

x is a numeric vector or a matrix. It represents the values of a cluster.

y is a numeric vector or a matrix. It represents the values of a cluster.

Details

This function is part of the hierarchical clusterization method. The function calculates the Chebyshev distance value from x and y.

Value

Chebyshev distance value.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
x <- c(1,2)
y <- c(1,3)

cluster1 <- matrix(x,ncol=2)
cluster2 <- matrix(y,ncol=2)

chebyshevDistance(x,y)
chebyshevDistance(cluster1,cluster2)
```

Description

To show the formula of the Chebyshev distance.

Usage

`chebyshevDistance.details(x, y)`
Arguments

- `x` is a numeric vector or a matrix. It represents the values of a cluster.
- `y` is a numeric vector or a matrix. It represents the values of a cluster.

Details

This function is part of the hierarchical clusterization method. The function calculates the Chebyshev distance value from `x` and `y`.

Value

Chebyshev distance value and formula.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
x <- c(1,2)
y <- c(1,3)
cluster1 <- matrix(x, ncol=2)
cluster2 <- matrix(y, ncol=2)
chebyshevDistance(x, y)
chebyshevDistance(cluster1, cluster2)
```

Description

To calculate the Chebyshev distance between clusters applying weights given.

Usage

`chebyshevDistanceW(cluster1, cluster2, weight)`
Arguments

cluster1 is a cluster.
cluster2 is a cluster.
weight is a numeric vector.

Details

The function calculates the Chebyshev distance value from cluster1 and cluster2, applying weights to the cluster's components.

Value

Chebyshev distance applying weights value.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)

weight1 <- c(0.4,0.6)
weight2 <- c(2,12)

chebyshevDistanceW(cluster1,cluster2,weight1)
chebyshevDistanceW(cluster1,cluster2,weight2)

Description

To explain how to calculate the Chebyshev distance between clusters applying weights given.

Usage

chebyshevDistanceW.details(cluster1, cluster2, weight)
clusterDistance

Arguments

- `cluster1` is a cluster.
- `cluster2` is a cluster.
- `weight` is a numeric vector.

Details

The function calculates the Chebyshev distance value from `cluster1` and `cluster2`, applying weights to the cluster's components.

Value

Chebyshev distance applying weights value. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)
weight1 <- c(0.4,0.6)
weight2 <- c(2,12)
chebyshevDistanceW.details(cluster1,cluster2,weight1)
chebyshevDistanceW.details(cluster1,cluster2,weight2)
```

clusterDistance  To calculate the distance between clusters.

Description

To calculate the distance between clusters depending on the approach and distance type.

Usage

`clusterDistance(cluster1, cluster2, approach, distance)`
clusterDistance.details

Arguments

- `cluster1` is a matrix
- `cluster2` is a matrix
- `approach` is a string. Type of function to apply.
- `distance` is a string. Type of distance to use.

Details

This function is part of the hierarchical clusterization method. The function calculates the final distance between `cluster1` and `cluster2` applying the approach definition, using the distance type given.

`approach` indicates the algorithm used to get the value. `distance` indicates the distance used to get the value. Possible values: 'MAX', 'MIN', 'AVG'.

Value

Distance between clusters.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,4),ncol=2)
clusterDistance.details(cluster1,cluster2,'AVG','MAX')
clusterDistance.details(cluster1,cluster2,'MAX','OCT')
```

Description

To explain how to calculate the distance between clusters depending on the approach and distance type.

Usage

```r
clusterDistance.details(cluster1, cluster2, approach, distance)
```
clusterDistanceByApproach

Arguments

- cluster1 is a matrix
- cluster2 is a matrix
- approach is a string. Type of function to apply.
- distance is a string. Type of distance to use.

Details

This function is part of the hierarchical clusterization method. The function explains how to calculate the final distance between cluster1 and cluster2 applying the approach definition, using the distance type given.

- approach indicates the algorithm used to get the value.
- distance indicates the distance used to get the value. Possible values: 'MAX', 'MIN', 'AVG'.

Value

Distance between clusters. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,4),ncol=2)
clusterDistance.details(cluster1,cluster2,'AVG','MAN')
clusterDistance.details(cluster1,cluster2,'MAX','OCT')
```

---

clusterDistanceByApproach

To calculate the distance by approach option.

Description

To calculate the distance depending on option given.

Usage

```r
clusterDistanceByApproach(distances, approach)
```
clusterDistanceByApproach.details

Arguments

- `distances` is a numeric vector.
- `approach` is a string. Type of function to apply.

Details

This function is part of the hierarchical clusterization method. The function calculates the distance value from `distances`.

`approach` indicates the algorithm used to get the value. Possible values: 'MAX', 'MIN', 'AVG'.

Value

- max, min or average from a vector.

Author(s)

- Roberto Alcántara <roberto.alcantara@edu.uah.es>
- Juan José Cuadrado <jjcg@uah.es>
- Universidad de Alcalá de Henares

Examples

```r
distances1 <- c(4,14,24,34)
distances2 <- c(1:10)
clusterDistanceByApproach(distances1,'MAX')
clusterDistanceByApproach(distances2,'MIN')
```

Description

To explain how to calculate the distance by approach option.

Usage

```r
clusterDistanceByApproach.details(distances, approach)
```

Arguments

- `distances` is a numeric vector.
- `approach` is a string. Type of function to apply.
complementaryClusters

Details

This function is part of the hierarchical clusterization method. The function explains how to calculate the distance value from distances.

approach indicates the algorithm used to get the value. Possible values: 'MAX', 'MIN', 'AVG'.

Value

max, min or average from a vector.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

distances1 <- c(4,14,24,34)
distances2 <- c(1:10)
clusterDistanceByApproach(distances1,'MAX')
clusterDistanceByApproach(distances2,'MIN')

_______________________________________________________________
complementaryClusters  To check if two clusters are complementary

Description

To check if two clusters include every element but without repeating anyone.

Usage

complementaryClusters(components, cluster1, cluster2)

Arguments

components is an elements list. It contains every component that has to be in one cluster or in the other one. But each element can only be included in one cluster.
cluster1 is a cluster (matrix).
cluster2 is a cluster (matrix).
Details
This function checks if the cluster that will be divided contains the simple elements that they have to include. They have to contain every element, but anyone should be duplicated.
The function will return a boolean value.

Value
Boolean value.

Author(s)
Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

data <- c(1,2,1,3,1,4,1,5)
components <- toListDivisive(data)
cluster1 <- matrix(c(1,2,1,3),ncol=2)
cluster2 <- matrix(c(1,4,1,5),ncol=2)
cluster3 <- matrix(c(1,6,1,7),ncol=2)
complementaryClusters(components,cluster1,cluster2) #TRUE
complementaryClusters(components,cluster3,cluster2) #FALSE

Description
To explain how and why two clusters are complementary.

Usage
complementaryClusters.details(components, cluster1, cluster2)

Arguments
components is an elements list. It contains every component that has to be in one cluster or in the other one. But each element can only be included in one cluster.
cluster1 is a cluster (matrix).
cluster2 is a cluster (matrix).
correlationHC

Details

This function checks if the cluster that will be divided contains the simple elements that they have to include. They have to contain every element, but anyone should be duplicated.

The function will return a boolean value.

Value

Boolean value. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

data <- c(1,2,1,3,1,4,1,5)
components <- toListDivisive(data)
cluster1 <- matrix(c(1,2,1,3),ncol=2)
cluster2 <- matrix(c(1,4,1,5),ncol=2)
cluster3 <- matrix(c(1,6,1,7),ncol=2)
complementaryClusters.details(components,cluster1,cluster2) #TRUE
complementaryClusters.details(components,cluster3,cluster2) #FALSE

correlationHC

To execute hierarchical correlation algorithm.

Description

To execute hierarchical correlation algorithm applying weights, distance types, ...

Usage

correlationHC(
  data, target = NULL,
  weight = c(),
  distance = "EUC",
  normalize = TRUE,
  labels = NULL
)
correlationHC

### Arguments

- **data**: is a data frame with the main data.
- **target**: is a data frame, a numeric vector or a matrix. Default value = NULL.
- **weight**: is a numeric vector. Default value = empty vector.
- **distance**: is a string. The distance type. Default value = Euclidean distance.
- **normalize**: is a boolean parameter. If the user wants to normalize weights. Default value = TRUE.
- **labels**: is a string vector. For the graphical solution. Default value = NULL.

### Details

This function execute the complete hierarchical correlation method.

1. The function transforms data in useful object to be used.
2. It creates the clusters.
3. It calculates the distance from the target to every cluster applying distance type given.
4. It orders the distance in increasing way.
5. It orders the clusters according to their distance from the previous step
6. It shows the clusters sorted and the distance used.

### Value

R object with a dendrogram, the sorted distances and the list with every cluster.

### Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

### Examples

```r
data <- matrix(c(1,2,1,4,5,1,8,2,9,6,3,5,8,5,4),ncol= 3)
dataFrame <- data.frame(data)
target1 <- c(1,2,3)
target2 <- dataFrame[1,]
weight1 <- c(1,6,3)
weight2 <- c(0.1,0.6,0.3)
correlationHC(dataFrame, target1)
```
correlationHC.details

To explain how hierarchical correlation algorithm works.

Description

To explain how the hierarchical correlation algorithm works.

Usage

```r
correlationHC.details(
  data,
  target = NULL,
  weight = c(),
  distance = "EUC",
  normalize = TRUE,
  labels = NULL
)
```

Arguments

data is a data frame with the main data.
target is a data frame, a numeric vector or a matrix. Default value = NULL.
weight is a numeric vector. Default value = empty vector.
distance is a string. The distance type. Default value = Euclidean distance.
normalize is a boolean parameter. If the user wants to normalize weights. Default value = TRUE.
labels is a string vector. For the graphical solution. Default value = NULL.

Details

This function explains the complete hierarchical correlation method. It explains the theoretical algorithm step by step.

1 - The function transforms data in useful object to be used.
2 - It creates the clusters.
3 - It calculates the distance from the target to every cluster applying the distance type given.
4 - It orders the distance in an increasing way.
5 - It orders the clusters according to their distance from the previous step.
6 - It shows the clusters sorted and the distance used.
distances

Value

R object with a dendrogram, the sorted distances and the list with every cluster. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

data <- matrix(c(1,2,1,4,5,1,8,2,9,6,3,5,8,5,4),ncol= 3)
dataFrame <- data.frame(data)
target1 <- c(1,2,3)
target2 <- dataFrame[1,]
weight1 <- c(1,6,3)
weight2 <- c(0.1,0.6,0.3)
correlationHC.details(dataFrame, target1)
correlationHC.details(dataFrame, target1, weight1)
correlationHC.details(dataFrame, target1, weight1, normalize = FALSE)
correlationHC.details(dataFrame, target1, weight2, 'CAN', FALSE)

distances

To calculate distances applying weights.

Description

To calculate distances between two clusters applying weights depending on the distance type.

Usage

distances(cluster1, cluster2, distance, weight)
Arguments

cluster1  is a matrix.
cluster2  is a matrix.
distance is a string. The distance type to apply.
weight is a numeric vector.

Details

This function calculates distance applying distance type and applying each weight to its characteristic.

Distance type could be EUC, MAN, CAN, CHE or OCT.

Value

Distance value applying weights.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
cluster1 <- matrix(c(2,3))
cluster2 <- matrix(c(4,5))

weight1 <- c(0.6,0.4)
weight2 <- c(2,4)

distances(cluster1, cluster2, 'MAN', weight1)
distances(cluster1, cluster2, 'CHE', weight2)
```

Description

To calculate distances applying weights.

Usage

distances.details(cluster1, cluster2, distance, weight)
Arguments

cluster1 is a matrix.
cluster2 is a matrix.
distance is a string. The distance type to apply.
weight is a numeric vector.

Details

This function calculates distance applying distance type and applying each weight to its characteristic.
Distance type could be EUC, MAN, CAN, CHE or OCT.

Value

Distance value applying weights. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

cluster1 <- matrix(c(2,3))
cluster2 <- matrix(c(4,5))
weight1 <- c(0.6,0.4)
weight2 <- c(2,4)
distances.details(cluster1, cluster2, 'MAN', weight1)
distances.details(cluster1, cluster2, 'CHE', weight2)

divisiveHC To execute divisive hierarchical clusterization algorithm by distance and approach.

Description

To execute complete divisive hierarchical clusterization algorithm by choosing distance and approach types.
**divisiveHC**

Usage

```
divisiveHC(data, distance, approach)
```

Arguments

- **data**: could be a numeric vector, a matrix or a numeric data frame. It will be transformed into matrix and list to be used.
- **distance**: is a string. It chooses the distance to use.
- **approach**: is a string. It chooses the approach to use.

Details

This function is the main part of the divisive hierarchical clusterization method. It executes the theoretical algorithm step by step.

1 - The function transforms data in useful object to be used.
2 - It creates a cluster that includes every simple elements.
3 - It initializes posible clusters using the initial elements.
4 - It calculates a matrix distance with the clusters created in the 3rd step.
5 - It chooses the maximal distance value and gets the clusters to be divided.
6 - It divides the cluster into two new complementary clusters and updates the clusters list.
6 - It repeats these steps until every cluster can’t be divided again. The solution includes every simple cluster.

Value

A list with the divided clusters.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```
a <- c(1,2,1,3,1,4,1,5,1,6)
matrixA <- matrix(a,ncol=2)
dataFrameA <- data.frame(matrixA)
divisiveHC(a,'EUC','MAX')
divisiveHC(matrixA,'MAN','AVG')
```
divisiveHC.details

To explain the divisive hierarchical clusterization algorithm by distance and approach.

Description
To explain the complete divisive hierarchical clusterization algorithm by choosing distance and approach types.

Usage

```r
divisiveHC.details(data, distance, approach)
```

Arguments

- `data` could be a numeric vector, a matrix or a numeric data frame. It will be transformed into matrix and list to be used.
- `distance` is a string. It chooses the distance to use.
- `approach` is a string. It chooses the approach to use.

Details

This function is the main part of the divisive hierarchical clusterization method. It explains the theoretical algorithm step by step.

1 - The function transforms data in useful object to be used.
2 - It creates a cluster that includes every simple elements.
3 - It initializes posible clusters using the initial elements.
4 - It calculates a matrix distance with the clusters created in the 3rd step.
5 - It chooses the maximal distance value and gets the clusters to be divided.
6 - It divides the cluster into two new complementary clusters and updates the clusters list.
6 - It repeats these steps until every cluster can’t be divided again. The solution includes every simple cluster.

Value

A list with the divided clusters. Explanation

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
edistance

Examples

```r
a <- c(1,2,1,3,1,4,1,5,1,6)
matrixA <- matrix(a,ncol=2)
dataFrameA <- data.frame(matrixA)
divisiveHC.details(a,'EUC','MAX')
divisiveHC.details(matrixA,'MAN','AVG')
divisiveHC.details(dataFrameA,'CHE','MIN')
```

---

edistance  To calculate the Euclidean distance.

Description

To calculate the Euclidean distance of two clusters.

Usage

```r
edistance(x, y)
```

Arguments

- `x` is a numeric vector or a matrix. It represents the values of a cluster.
- `y` is a numeric vector or a matrix. It represents the values of a cluster.

Details

This function is part of the hierarchical clusterization method. The function calculates the Euclidean distance value from `x` and `y`.

Value

Euclidean distance value.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
**Examples**

```r
x <- c(1,2)
y <- c(1,3)
cluster1 <- matrix(x,ncol=2)
cluster2 <- matrix(y,ncol=2)
edistance(x,y)
edistance(cluster1,cluster2)
```

**Description**

To show the Euclidean distance formula and to calculate the Euclidean distance of two clusters.

**Usage**

```r
edistance.details(x, y)
```

**Arguments**

- `x` is a numeric vector or a matrix. It represents the values of a cluster.
- `y` is a numeric vector or a matrix. It represents the values of a cluster.

**Details**

This function is part of the hierarchical clusterization method. The function calculates the Euclidean distance value from `x` and `y`.

**Value**

Euclidean distance value and formula.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

\begin{verbatim}
x <- c(1,2)
y <- c(1,3)
cluster1 <- matrix(x, ncol=2)
cluster2 <- matrix(y, ncol=2)
edistance(x,y)
edistance(cluster1,cluster2)
\end{verbatim}

Description

To calculate the Euclidean distance between clusters applying weights given.

Usage

\texttt{edistanceW(cluster1, cluster2, weight)}

Arguments

- \texttt{cluster1} is a cluster.
- \texttt{cluster2} is a cluster.
- \texttt{weight} is a numeric vector.

Details

The function calculates the Euclidean distance value from cluster1 and cluster2, applying weights to the cluster's components.

Value

Euclidean distance applying weights value.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)
weight1 <- c(0.4,0.6)
weight2 <- c(2,12)
edistanceW(cluster1,cluster2,weight1)
edistanceW(cluster1,cluster2,weight2)
```

Description

To explain how to calculate the Euclidean distance between clusters applying weights given.

Usage

```r
edistanceW.details(cluster1, cluster2, weight)
```

Arguments

- `cluster1` is a cluster.
- `cluster2` is a cluster.
- `weight` is a numeric vector.

Details

The function calculates the Euclidean distance value from `cluster1` and `cluster2`, applying weights to the cluster's components.

Value

Euclidean distance applying weights value. Explanation.

Author(s)

- Roberto Alcántara <roberto.alcantara@edu.uah.es>
- Juan José Cuadrado <jjcg@uah.es>
- Universidad de Alcalá de Henares
**getCluster**

**Examples**

```r
cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)

weight1 <- c(0.4,0.6)
weight2 <- c(2,12)

edistanceW.details(cluster1,cluster2,weight1)
edistanceW.details(cluster1,cluster2,weight2)
```

---

**Description**

To get the clusters with minimal distance value. By using the given distance, it gets the matrix index.

**Usage**

```r
getCluster(distance, matrix)
```

**Arguments**

- `distance` is a number. It should be in the matrix.
- `matrix` is a numeric matrix.

**Details**

This function is part of the hierarchical clusterization method. The function uses the distance value and gets the clustersId with the minimal distance.

For the divisive algorithm, it chooses the distances from a distances list.

**Value**

numeric vector with two cluster indexes.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
matrixExample <- matrix(c(1:10), ncol=2)
getCluster(2,matrixExample)
```

---

**getCluster.details**  
To explain how to get the clusters with minimal distance.

**Description**
To explain how to get the clusters with the minimal distance value. By using the given distance, it gets the matrix index.

**Usage**

```r
getCluster.details(distance, matrix)
```

**Arguments**
- `distance` is a number. It should be in the matrix.
- `matrix` is a numeric matrix.

**Details**
This function is part of the hierarchical clusterization method. The function uses the distance value and gets the clustersId with the minimal distance. For the divisive algorithm, it chooses the distances from a distances list.

**Value**
Numeric vector with two clusters indexes.

**Author(s)**
Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

**Examples**

```r
matrixExample <- matrix(c(1:10), ncol=2)
getCluster.details(2,matrixExample)
```
getClusterDivisive

To get the clusters with maximal distance.

Description

To get the clusters with the maximal distance value. By using the given distance, it gets the matrix index.

Usage

getClusterDivisive(distance, vector)

Arguments

distance  is a number. It should be in the matrix.
vector    is a numeric vector

Details

This function is part of the hierarchical clusterization method. The function uses the distance value and gets the clustersId with the minimal distance.

For the divisive algorithm, it chooses the distances from a distances list.

Value

A cluster.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

getClusterDivisive(2,c(1:10))
getClusterDivisive(6,c(2,4,6,8,10,12))
getClusterDivisive.details

To explain how to get the clusters with maximal distance.

Description

To explain how to get the clusters with the maximal distance value. By using the given distance, it gets the matrix index.

Usage

getClusterDivisive.details(distance, vector)

Arguments

distance is a number. It should be in the matrix.
vector is a numeric vector.

Details

This function is part of the hierarchical clusterization method. The function uses the distance value and gets the clustersId with the minimal distance.
For the divisive algorithm, it chooses the distances from a distances list.

Value

A cluster. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

getClusterDivisive.details(2,c(1:10))
getClusterDivisive(6,c(2,4,6,8,10,12))
initClusters  To initialize clusters for the divisive algorithm.

Description
To initialize clusters for the divisive algorithm.

Usage
initClusters(initList)

Arguments
initList is a clusters list. It will contain clusters with one element.

Details
This function will calculate every cluster that can be created by joining initial clusters with each other. It creates clusters from length = 1 until a cluster with every element is created.

These clusters will be used to find the most different clusters that we can create by dividing the initial cluster.

Value
A cluster list.

Author(s)
Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

data <- c(1:8)
matrix <- matrix(data,ncol=2)
listData <- toListDivisive(data)
listMatrix <- toListDivisive(matrix)
initClusters(listData)
initClusters(listMatrix)
Description

To explain how to initialize clusters for the divisive algorithm.

Usage

initClusters.details(initList)

Arguments

initList is a clusters list. It will contain clusters with one element.

Details

This function will explain how to calculate every cluster that can be created by joining initial clusters with each other. It creates clusters from length = 1 until a cluster with every element is created. These clusters will be used to find the most different clusters that we can create by dividing the initial cluster.

Value

A cluster list. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

data <- c(1:8)
matrix <- matrix(data, ncol=2)
listData <- toListDivisive(data)
listMatrix <- toListDivisive(matrix)
initClusters.details(listData)
initClusters.details(listMatrix)
**initData**  

To initialize data, hierarchical correlation algorithm.

**Description**

To initialize data, hierarchical correlation algorithm.

**Usage**

`initData(data)`

**Arguments**

- `data` is a data frame with the main data.

**Details**

This function is part of the hierarchical correlation method. The function initializes `data` transforming each row from the data frame into a matrix with every row elements.

**Value**

A cluster list. Initializing data.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>  
Juan José Cuadrado <jjcg@uah.es>  
Universidad de Alcalá de Henares

**Examples**

```r
data <- matrix(c(1,2,1,4,5,1,8,2,9,6,3,5,8,5,4), ncol=3)
dataFrame <- data.frame(data)
initData(dataFrame)
```
**initData.details**

To initialize data, hierarchical correlation algorithm.

**Description**

To explain how to initialize data, hierarchical correlation algorithm.

**Usage**

```r
initData.details(data)
```

**Arguments**

- `data` is a data frame with the main data.

**Details**

This function is part of the hierarchical correlation method. The function initializes `data` transforming each row from the data frame into a matrix with every row elements.

**Value**

A cluster list. Initializing data. Explanation.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

**Examples**

```r
data <- matrix(c(1,2,1,4,5,1,8,2,9,6,3,5,8,5,4), ncol = 3)
dataFrame <- data.frame(data)
initData.details(dataFrame)
```
initImages  

To display an image.

**Description**

An auxiliary function to display a picture.

**Usage**

`initImages(path)`

**Arguments**

- `path` is a file path.

initTarget  

To initialize target, hierarchical correlation algorithm.

**Description**

To initialize target, hierarchical correlation algorithm. It checks if target is valid, if not, it initializes the target.

**Usage**

`initTarget(target, data)`

**Arguments**

- `target` is a numeric vector, a matrix or a data frame.
- `data` is a data frame with the main data.

**Details**

This function is part of the hierarchical correlation method. The function initializes `target` and checks if it is a valid target.

The function transforms the target into a matrix. Then, it checks if the target has only one row and the same columns has the main data.

If it is not a valid target, the function will notice the problem and will initialize a new target with every column with value 0.

**Value**

A cluster.
Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
data <- matrix(c(1,2,1,4,5,1,8,2,9,6,3,5,8,5,4),ncol= 3)
dataFrame <- data.frame(data)
target1 <- matrix(c(2,3))
target2 <- matrix(c(2,3,6))
initTarget(target1,dataFrame)
initTarget(target2,dataFrame)
```

initTarget.details

To initialize target, hierarchical correlation algorithm.

Description

To initialize target, hierarchical correlation algorithm. It checks if target is valid, if not, it initializes the target.

Usage

```r
initTarget.details(target, data)
```

Arguments

- `target` is a numeric vector, a matrix or a data frame.
- `data` is a data frame with the main data.

Details

This function is part of the hierarchical correlation method. The function initializes `target` and checks if it is an acceptable target.

The function transforms the target into a matrix. Then, it checks if the target has only one row and the same columns have the main data.

If it is not an acceptable target, the function will notice the problem and will initialize a new target with every column with value 0.
Value

A cluster. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

data <- matrix(c(1,2,1,4,5,1,8,2,9,6,3,5,8,5,4),ncol= 3)
dataFrame <- data.frame(data)
target1 <- matrix(c(2,3))
target2 <- matrix(c(2,3,6))
initTarget.details(target1,dataFrame)
initTarget.details(target2,dataFrame)

matrixDistance

Matrix distance by distance type

Description

To calculate the matrix distance by using distance type.

Usage

matrixDistance(list, distance)

Arguments

list is a clusters list.
distance is a literal.

Details

This function is part of the hierarchical clusterization method. The function calculates the matrix distance by using the distance type given.

The list parameter will be a list with the clusters as rows and columns.
The function avoids distances equal 0 and undefined clusters.
maxDistance

Examples

```r
data <- c(1:10)
clusters <- toList(data)
matrixDistance(clusters, 'EUC')
```

```
maxDistance(matrixExample)
```

Description

Get the matrix maximal value.

Usage

```r
maxDistance(matrix)
```

Arguments

- `matrix` is a numeric matrix. It could be a numeric vector.

Details

This function is part of the hierarchical clusterization method. The function uses the numeric vector or matrix `matrix` given and return the maximal value. The function avoids distances equal 0, and initialize maximal value with an auxiliary function `initMax`, which gets the first matrix element with a valid distance.

Value

Numeric value. Max value from a matrix

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples

```r
matrixExample <- matrix(c(1:10), nrow=2)
maxDistance(1:10)
maxDistance(matrixExample)
```
**maxDistance.details**  

**Maximal distance**

**Description**

To explain how to get the matrix maximal value.

**Usage**

```r
maxDistance.details(matrix)
```

**Arguments**

- `matrix` is a numeric matrix. It could be a numeric vector.

**Details**

This function is part of the hierarchical clusterization method. The function uses the numeric vector or matrix `matrix` given and return the maximal value. The function avoids distances equal 0, and initialize maximal value with an auxiliar function `initMax`, which gets the first matrix element with a valid distance.

**Value**

Numeric value. Max value from a matrix. Explanation.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

**Examples**

```r
matrixExample <- matrix(c(1:10), nrow=2)
maxDistance.details(1:10)
maxDistance.details(matrixExample)
```
**Description**

To calculate the matrix distance by using distance and approach type.

**Usage**

```r
mdAgglomerative(list, distance, approach)
```

**Arguments**

- `list` is a clusters list.
- `distance` is a literal. The distance type to be used.
- `approach` is a literal. The approach type to be used.

**Details**

This function is part of the hierarchical clusterization method. The function calculates the matrix distance by using the distance and approach type given.

The `list` parameter will be a list with the clusters as rows and columns.

The function avoids distances equal 0 and undefined clusters.

**Value**

A matrix distance.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

**Examples**

```r
data <- c(1,2,1,3,1,4,1,5,1,6)
clusters <- tolist(data)
mdAgglomerative(clusters, 'EUC', 'MAX')
mdAgglomerative(clusters, 'CHE', 'AVG')
```
*mdAgglomerative.details*

Matrix distance by distance and approach type.

**Description**

To explain how to calculate the matrix distance by using distance and approach type.

**Usage**

```r
mdAgglomerative.details(list, distance, approach)
```

**Arguments**

- `list` is a clusters list.
- `distance` is a literal. The distance type to be used.
- `approach` is a literal. The approach type to be used.

**Details**

This function is part of the hierarchical clusterization method. The function calculates the matrix distance by using the distance and approach type given.

The `list` parameter will be a list with the clusters as rows and columns.

The function avoids distances equal 0 and undefined clusters.

**Value**

A matrix distance. Explanation.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>

Universidad de Alcalá de Henares

**Examples**

```r
data <- c(1,2,1,3,1,4,1,5,1,6)
clusters <- toList(data)
mdAgglomerative.details(clusters, 'EUC', 'MAX')
mdAgglomerative.details(clusters, 'CHE', 'AVG')
```
Description
To calculate the matrix distance by using distance and approach types.

Usage
mdDivisive(list, distance, approach, components)

Arguments
- list: is a clusters list.
- distance: is a string. The distance type to be used.
- approach: is a string. The approach type to be used.
- components: is a clusters list. It contains every clusters with only one element. It is used to check if complementary condition is 'TRUE'.

Details
This function is part of the divisive hierarchical clusterization method. The function calculates the matrix distance by using the distance and approach types given.
- The list parameter will be a list with the clusters as rows and columns.
- The function avoids distances equal 0 and undefined clusters.
- It also avoids distances between clusters that are not complementary because they can’t be chosen to divide all the clusters.

Value
Matrix distance.

Author(s)
Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares

Examples
```
data <- c(1,2,1,3,1,4,1,5,1,6)
clusters <- toList(data)
components <- toList(data)
```
\texttt{mdDivisive.details}

\begin{verbatim}
mdDivisive(clusters, 'EUC', 'MAX', components)
mdDivisive(clusters, 'MAN', 'MIN', components)
\end{verbatim}

---

**Description**

To explain how to calculate the matrix distance by using distance and approach types.

**Usage**

\texttt{mdDivisive.details(list, distance, approach, components)}

**Arguments**

- \texttt{list} is a clusters list.
- \texttt{distance} is a string. The distance type to be used.
- \texttt{approach} is a string. The approach type to be used.
- \texttt{components} is a clusters list. It contains every clusters with only one element. It is used to check if complementary condition is 'TRUE'.

**Details**

This function is part of the divisive hierarchical clusterization method. The function calculates the matrix distance by using the distance and approach types given.

The \texttt{list} parameter will be a list with the clusters as rows and columns.

The function avoids distances equal 0 and undefined clusters.

It also avoids distances between clusters that are not complementary because they can’t be chosen to divide all the clusters.

**Value**

Matrix distance. Explanation.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

data <- c(1,2,1,3,1,4,1,5,1,6)
clusters <- toList(data)
components <- toList(data)
mdDivisive.details(clusters, 'EUC', 'MAX', components)
mdDivisive.details(clusters, 'MAN', 'MIN', components)

---

mdistance

To calculate the Manhattan distance.

Description
To calculate the Manhattan distance of two clusters.

Usage
mdistance(x, y)

Arguments
x is a numeric vector or a matrix. It represents the values of a cluster.
y is a numeric vector or a matrix. It represents the values of a cluster.

Details
This function is part of the hierarchical clusterization method. The function calculates the Manhattan distance value from x and y.

Value
Manhattan distance value.

Author(s)
Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
x <- c(1,2)
y <- c(1,3)
cluster1 <- matrix(x,ncol=2)
cluster2 <- matrix(y,ncol=2)
mdistance(x,y)
mdistance(cluster1,cluster2)
```

Description

To explain how to calculate the Manhattan distance of two clusters.

Usage

```r
mdistance.details(x, y)
```

Arguments

- `x` is a numeric vector or a matrix. It represents the values of a cluster.
- `y` is a numeric vector or a matrix. It represents the values of a cluster.

Details

This function is part of the hierarchical clusterization method. The function calculates the Manhattan distance value from `x` and `y`.

Value

Manhattan distance value and formula.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
**Examples**

```r
x <- c(1,2)
y <- c(1,3)
cluster1 <- matrix(x,ncol=2)
cluster2 <- matrix(y,ncol=2)
mdistance(x,y)
mdistance(cluster1,cluster2)
```

---

**mdistanceW**

To calculate the Manhattan distance applying weights.

---

**Description**

To calculate the Manhattan distance between clusters applying weights given.

**Usage**

```r
mdistanceW(cluster1, cluster2, weight)
```

**Arguments**

- `cluster1` is a cluster.
- `cluster2` is a cluster.
- `weight` is a numeric vector.

**Details**

The function calculates the Manhattan distance value from `cluster1` and `cluster2`, applying weights to the cluster's components.

**Value**

Manhattan distance applying weights value.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)

weight1 <- c(0.4,0.6)
weight2 <- c(2,12)

mdistanceW(cluster1, cluster2, weight1)
mdistanceW(cluster1, cluster2, weight2)
```

**Description**

To explain how to calculate the Manhattan distance between clusters applying weights given.

**Usage**

```r
mdistanceW.details(cluster1, cluster2, weight)
```

**Arguments**

- `cluster1` is a cluster.
- `cluster2` is a cluster.
- `weight` is a numeric vector.

**Details**

The function calculates the Manhattan distance value from `cluster1` and `cluster2`, applying weights to the cluster’s components.

**Value**

Manhattan distance applying weights value. Explanation.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

cluster1 <- matrix(c(1,2),ncol=2)
class2 <- matrix(c(1,3),ncol=2)

weight1 <- c(0.4,0.6)
weight2 <- c(2,12)

mdistanceW.details(cluster1,cluster2,weight1)
mdistanceW.details(cluster1,cluster2,weight2)

<table>
<thead>
<tr>
<th>minDistance</th>
<th>Minimal distance</th>
</tr>
</thead>
</table>

Description

Get the matrix minimal value.

Usage

minDistance(matrix)

Arguments

matrix is a numeric matrix. It could be a numeric vector.

Details

This function is part of the hierarchical clusterization method. The function uses the numeric vector or matrix matrix given and return the minimal value. The function avoids distances equal 0, and initialize minimum value with an auxiliar function initMin, which gets the first matrix element with a valid distance.

Value

Numeric value. Min value from a matrix.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
matrixExample <- matrix(c(1:10), nrow=2)
minDistance(1:10)
minDistance(matrixExample)
```

Description

To explain how to get the matrix minimal value.

Usage

```r
minDistance.details(matrix)
```

Arguments

- `matrix` is a numeric matrix. It could be a numeric vector.

Details

This function is part of the hierarchical clusterization method. The function uses the numeric vector or matrix `matrix` given and return the minimal value. The function avoids distances equal 0, and initialize minimum value with an auxiliar function `initMin`, which gets the first matrix element with a valid distance.

Value

Numeric value. Min value from a matrix. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

    matrixExample <- matrix(c(1:10), nrow=2)
    minDistance(1:10)
    minDistance.details(matrixExample)

---

newCluster

To create a new cluster.

Description

To create the cluster formed by the two clusters given. Add the new cluster to list.

Usage

    newCluster(list, clusters)

Arguments

    list is the generic cluster list.
    clusters is a vector with the matrix index clusters.

Details

This function is part of the hierarchical clusterization method.
1 - The function maps clusters in list.
2 - It creates a new cluster from them.
3 - It adds the new cluster to list.
4 - It disables the clusters used in the second step.

Value

A list with clusters.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
newCluster.details

Examples

```r
data <- c(1:10)
list <- toList(data)
clusters <- c(1,2)
newCluster(list,clusters)
```

newCluster.details

To explain how to create a new cluster.

Description

To explain how to create the cluster formed by the two clusters given. Add the new cluster to `list`.

Usage

`newCluster.details(list, clusters)`

Arguments

- `list` is the generic cluster list.
- `clusters` is a vector with the matrix index clusters.

Details

This function is part of the hierarchical clusterization method.

1 - The function maps `clusters` in `list`.
2 - It creates a new cluster from them.
3 - It adds the new cluster to `list`.
4 - It disables the clusters used in the second step.

Value

A list with clusters. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

data <- c(1:10)
list <- toList(data)
clusters <- c(1,2)
newCluster.details(list,clusters)

normalizeWeight

To normalize weight values.

Description
To normalize weight values if normalize = TRUE.

Usage
normalizeWeight(normalize, weight, data)

Arguments
normalize  is a boolean value.
weight     is a numeric vector.
data       is a data.frame.

Details
This function allows users to normalize weights.
If there is not any weight, the function will create a numeric vector of "1".
If normalize = TRUE, the function will make every weight value as a "[0:1]" value.
If normalize = FALSE, the function will not make any changes, weights will be the same.

Value
Numeric vector with updated weights.

Author(s)
Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
**normalizeWeight.details**

**Examples**

```r
data <- data.frame(matrix(c(1:10),ncol = 2))
weight1 <- c(0.6,0.4)
weight2 <- c(2,4)
normalizeWeight(FALSE, weight1, data)
normalizeWeight(TRUE, weight2, data)
normalizeWeight(FALSE, weight2, data)
```

**normalizeWeight.details**

To normalize weight values.

**Description**

To explain how to normalize weight values if `normalize` = TRUE.

**Usage**

`normalizeWeight.details(normalize, weight, data)`

**Arguments**

- `normalize` is a boolean value.
- `weight` is a numeric vector.
- `data` is a data.frame.

**Details**

This function allows users to normalize weights.

If there is not any weight, the function will create a numeric vector of "1".

If `normalize` = TRUE, the function will make every weight value as a "[0:1]" value.

If `normalize` = FALSE, the function will not make any changes, weights will be the same.

**Value**

Numeric vector with updated weights. Explanation.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
**octileDistance**

**Examples**

```r
data <- data.frame(matrix(c(1:10),ncol = 2))
weight1 <- c(0.6,0.4)
weight2 <- c(2,4)
normalizeWeight.details(FALSE, weight1, data)
normalizeWeight.details(TRUE, weight2, data)
normalizeWeight.details(FALSE, weight2, data)
```

**octileDistance**  
*To calculate the Octile distance.*

**Description**

To calculate the octile distance of two clusters.

**Usage**

```r
octileDistance(x, y)
```

**Arguments**

- `x`  
  is a numeric vector or a matrix. It represents the values of a cluster.

- `y`  
  is a numeric vector or a matrix. It represents the values of a cluster.

**Details**

This function is part of the hierarchical clusterization method. The function calculates the octile distance value from `x` and `y`.

**Value**

Octile distance value.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
x <- c(1,2)
y <- c(1,3)

c1 <- matrix(x,ncol=2)
c2 <- matrix(y,ncol=2)

octileDistance(x,y)
octileDistance(c1,c2)
```

Description

To explain how to calculate the Octile distance.

Usage

```r
octileDistance(x, y)
```

Arguments

- `x` is a numeric vector or a matrix. It represents the values of a cluster.
- `y` is a numeric vector or a matrix. It represents the values of a cluster.

Details

This function is part of the hierarchical clusterization method. The function calculates the octile distance value from `x` and `y`.

Value

Octile distance value.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
x <- c(1,2)
y <- c(1,3)
cluster1 <- matrix(x,ncol=2)
cluster2 <- matrix(y,ncol=2)
octileDistance.details(x,y)
octileDistance.details(cluster1,cluster2)
```

---

**octileDistanceW**

To calculate the Octile distance applying weights.

**Description**

To calculate the Octile distance between clusters applying weights given.

**Usage**

```r
octileDistanceW(cluster1, cluster2, weight)
```

**Arguments**

- `cluster1` is a cluster.
- `cluster2` is a cluster.
- `weight` is a numeric vector.

**Details**

The function calculates the Octile distance value from `cluster1` and `cluster2`, applying weights to the cluster's components.

**Value**

Octile distance applying weights value.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```
cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)

weight1 <- c(0.4,0.6)
weight2 <- c(2,12)

octileDistanceW(cluster1,cluster2,weight1)
octileDistanceW(cluster1,cluster2,weight2)
```

Description

To explain how to calculate the Octile distance between clusters applying weights given.

Usage

```
octileDistanceW.details(cluster1, cluster2, weight)
```

Arguments

- `cluster1` is a cluster.
- `cluster2` is a cluster.
- `weight` is a numeric vector.

Details

The function calculates the Octile distance value from `cluster1` and `cluster2`, applying weights to the cluster’s components.

Value

Octile distance applying weights value. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
cluster1 <- matrix(c(1,2),ncol=2)
cluster2 <- matrix(c(1,3),ncol=2)

weight1 <- c(0.4,0.6)
weight2 <- c(2,12)

octileDistanceW.details(cluster1,cluster2,weight1)
octileDistanceW.details(cluster1,cluster2,weight2)
```

toList

To transform data into list

Description

To transform data into list.

Usage

```r
toList(data)
```

Arguments

data could be a numeric vector, a matrix or a numeric data frame.

Details

This function is part of the agglomerative hierarchical clusterization method. The function initializes data content as a list.

In agglomerative algorithm, it adds a TRUE flag to each element, which indicates that the cluster is not grouped.

Value

A list with clusters.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

data <- c(1:10)
matrix <- matrix(data,ncol=2)
dataFrame <- data.frame(matrix)
tolist(data)
tolist(matrix)
tolist(dataFrame)

tolist.details  To explain how to transform data into list

Description
To explain how to transform data into list.

Usage

tolist.details(data)

Arguments

data could be a numeric vector, a matrix or a numeric data frame.

Details
This function is part of the agglomerative hierarchical clusterization method. The function in-
itializes data content as a list.

In agglomerative algorithm, it adds a TRUE flag to each element, which indicates that the cluster is
not grouped.

Value
A list with clusters. Explanation.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
data <- c(1:10)
matrix <- matrix(data, ncol=2)
dataFrame <- data.frame(matrix)
tolist(data)
tolist(matrix)
tolist(dataFrame)
```

---

tolistDivisive

To transform data into list

Description

To transform data into list.

Usage

```r
tolistDivisive(data)
```

Arguments

data could be a numeric vector, a matrix or a numeric data frame.

Details

This function is part of the divisive hierarchical clusterization method. The function initializes data content as a list.

Value

a list with clusters.

Author(s)

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

```r
data <- c(1:10)
matrix <- matrix(data, ncol=2)
dataFrame <- data.frame(matrix)
toListDivisive(data)
toListDivisive(matrix)
toListDivisive(dataFrame)
```

**toListDivisive.details**

*To explain how to transform data into list*

**Description**

To explain how to transform data into list.

**Usage**

```r
toListDivisive.details(data)
```

**Arguments**

- `data` could be a numeric vector, a matrix or a numeric data frame.

**Details**

This function is part of the divisive hierarchical clusterization method. The function initializes data content as a list.

**Value**

A list with clusters. Explanation.

**Author(s)**

Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

data <- c(1:10)
matrix <- matrix(data,ncol=2)
dataFrame <- data.frame(matrix)
toListDivisive.details(data)
toListDivisive.details(matrix)
toListDivisive.details(dataFrame)

usefulClusters  To delete clusters grouped.

Description
To delete the clusters already used to create a new one.

Usage
usefulClusters(list)

Arguments
list is a list of clusters.

Details
This function is part of the hierarchical clusterization method. The function updates the cluster list with the clusters used after to calculate de matrix distance.

Value
A list of clusters.

Author(s)
Roberto Alcántara <roberto.alcantara@edu.uah.es>
Juan José Cuadrado <jjcg@uah.es>
Universidad de Alcalá de Henares
Examples

data <- c(1:10)
list <- toList(data)
usefulClusters(list)
Index

agglomerativeHC, 3
agglomerativeHC.details, 4

canberradistance, 5
canberradistance.details, 6
canberradistanceW, 7
canberradistanceW.details, 8
chebyshevDistance, 9
chebyshevDistance.details, 10
chebyshevDistanceW, 11
chebyshevDistanceW.details, 12
clusterDistance, 13
clusterDistance.details, 14
clusterDistanceByApproach, 15
clusterDistanceByApproach.details, 16
complementaryClusters, 17
complementaryClusters.details, 18
correlationHC, 19
correlationHC.details, 21
distances, 22
distances.details, 23
divisiveHC, 24
divisiveHC.details, 26
edistance, 27
edistance.details, 28
edistanceW, 29
edistanceW.details, 30
getCluster, 31
getCluster.details, 32
getClusterDivisive, 33
getClusterDivisive.details, 34

initClusters, 35
initClusters.details, 36
initData, 37
initData.details, 38
initImages, 39
initTarget, 39

initTarget.details, 40
matrixDistance, 41
maxDistance, 42
maxDistance.details, 43
mdAgglomerative, 44
mdAgglomerative.details, 45
mdDivisive, 46
mdDivisive.details, 47
mdistance, 48
mdistance.details, 49
mdistanceW, 50
mdistanceW.details, 51
minDistance, 52
minDistance.details, 53
newCluster, 54
newCluster.details, 55
normalizeWeight, 56
normalizeWeight.details, 57
octileDistance, 58
octileDistance.details, 59
octileDistanceW, 60
octileDistanceW.details, 61
toList, 62
toList.details, 63
toListDivisive, 64
toListDivisive.details, 65
usefulClusters, 66