Package 'multisom'

October 13, 2022

Type Package

Title Clustering a Data Set using Multi-SOM Algorithm

Version 1.3

Author Sarra Chair and Malika Charrad

Maintainer Sarra Chair <sarra.chair@gmail.com>

Description Implements two versions of the algorithm namely: stochastic and batch. The package determines also the best number of clusters and offers to the user the best clustering scheme from different results.

License GPL-2

Depends R (>= 3.1.3), class

Imports kohonen

URL https://sites.google.com/site/malikacharrad/research/multisom-package

NeedsCompilation no

Repository CRAN

Date/Publication 2017-05-23 17:28:23 UTC

R topics documented:

BatchSOM *Self-Organizing Map: Batch version*

Description

This function implements the batch version of the kohonen algorithm

Usage

```
BatchSOM(data,grid = somgrid(),min.radius=0.0001,
         max.radius=0.002,maxit=1000,
         init=c("random","sample","linear"),
         radius.type=c("gaussian","bubble","cutgauss","ep"))
```

Arguments

data	data to be used
grid	a grid for the representatives.The numbers of nodes should be approximately equal to 5*sqrt(n), which n denotes the number of sample.
min.radius	the minimum neighbourhood radius
max.radius	the maximum neighbourhood radius
maxit	the maximum number of iterations to be done
init	the method to be used to initialize the prototypes.The following are permitted: "random" uses random draws from N(0,1); "sample" uses a radom sample from the data; "linear" uses the linear grids upon the first two principle components direction.See package som.
radius.type	the neighborhood function type. The following are permitted: "gaussian" "bubble" "cutgauss" "ep"

Value

classif	a vector of integer indicating to which unit each observation has been assigned
codes	a matrix of code vectors
grid	the grid, an object of class "somgrid"

Author(s)

Sarra Chair and Malika Charrad

References

Kohonen, T. (1995) *Self-Organizing Maps.* Springer-Verlag.

Brian Ripley, William Venables (2015), class: Functions for Classification, URL https://cran.r-project.org/package=class.

Jun Yan (2010), som: Self-Organizing Map, URL https://cran.r-project.org/package=som.

Examples

```
data<-iris[,-c(5)]
BatchSOM(data,grid = somgrid(7,7,"hexagonal"),min.radius=0.0001,
             max.radius=0.002,maxit=1000,"random","gaussian")
```

`multisom.batch` *MultiSOM for batch version*

Description

This function implements the batch version of MultiSOM algorithm.

Usage

```
multisom.batch(data= NULL,xheight,xwidth,topo=c("rectangular",
           "hexagonal"),min.radius,max.radius,maxit=1000,
           init=c("random","sample","linear"),radius.type=
           c("gaussian","bubble","cutgauss","ep"),index="all")
```

Arguments

`data`	data to be used
`xheight`	the x-dimension of the map
`xwidth`	the y-dimension of the map
`topo`	the topology used to build the grid.The following are permitted: `"hexagonal"` `"rectangular"`
`min.radius`	the minimum neighbourhood radius
`max.radius`	the maximum neighbourhood radius
`maxit`	the maximum number of iterations to be done
`init`	the method to be used to initialize the prototypes.The following are permitted: `"random"` uses random draws from N(0,1); `"sample"` uses a radom sample from the data; `"linear"` uses the linear grids upon the first two principle components direction.

radius.type	the neighborhood function type. The following are permitted: "gaussian" "bubble" "cutgauss" "ep"
index	vector of the index to be calculated. This should be one of : "db", "dunn", "silhouette", "ptbiserial", "ch", "cindex", "ratkowsky", "mcclain", "gamma", "gplus", "tau", "ccc", "scott", "marriot", "trcovw", "tracew", "friedman", "rubin", "ball", "sdbw", "dindex", "hubert", "sv", "xie-beni", "hartigan", "ssi", "xu", "rayturi", "pbm", "banfeld", "all" (all indices will be used)

Details

Index	Optimal number of clusters
1. "db" or "all" (Davies and Bouldin 1979)	Minimum value of the index
2. "dunn" or "all" (Dunn 1974)	Maximum value of the index
3. "silhouette" or "all" (Rousseeuw 1987)	Maximum value of the index
4. "ptbiserial" or "all" (Milligan 1980, 1981)	Maximum value of the index
5. "ch" or "all" (Calinski and Harabasz 1974)	Maximum value of the index
6. "cindex" or "all" (Hubert and Levin 1976)	Minimum value of the index
7. "ratkowsky" or "all" (Ratkowsky and Lance 1978)	Maximum value of the index
8. "mcclain" or "all" (McClain and Rao 1975)	Minimum value of the index
9. "gamma" or "all" (Baker and Hubert 1975)	Maximum value of the index
10. "gplus" or "all" (Rohlf 1974) (Milligan 1981)	Minimum value of the index
11. "tau" or "all" (Rohlf 1974) (Milligan 1981)	Maximum value of the index
12. "ccc" or "all" (Sarle 1983)	Maximum value of the index
13. "scott" or "all" (Scott and Symons 1971)	Max. difference between hierarchy levels of the index
14. "marriot" or "all" (Marriot 1971)	Max. value of second differences between levels of the index
15. "trcovw" or "all" (Milligan and Cooper 1985)	Max. difference between hierarchy levels of the index
16. "tracew" or "all" (Milligan and Cooper 1985)	Max. value of absolute second differences between levels of the index
17. "friedman" or "all" (Friedman and Rubin 1967)	Max. difference between hierarchy levels of the index
18. "rubin" or "all" (Friedman and Rubin 1967)	Min. value of second differences between levels of the index
19. "ball" or "all"	Max. difference between hierarchy

(Ball and Hall 1965)	levels of the index
20. "sdbw" or "all"	Minimum value of the index
(Halkidi and Vazirgiannis 2001)	
21. "dindex" or "all"	Graphical method
(Lebart et al. 2000)	
22. "hubert" or "all"	Graphical method
(Hubert and Arabie 1985)	
23. "sv" or "all"	Maximum value of the index
(Zalik and Zalik, 2011)	
24. "xie-beni" or "all"	Minimum value of the index
(Xie and Beni 1991)	
25. "hartigan" or "all"	Maximum difference between
(Hartigan 1975)	hierarchy levels of the index
26. "ssi" or "all"	Maximum value of the index
(Dolnicar,Grabler and Mazanec 1999)	
27. "xu" or "all"	Max. value of second differences
(Xu 1997)	between levels of the index
28. "rayturi" or "all"	Minimum value of the index
(Ray and Turi 1999)	
29. "pbm" or "all"	Maximum value of the index
(Bandyopadhyay,Pakhira and Maulik 2004)	
30. "banfeld" or "all"	Minimum value of the index
(Banield and Raftery 1974)	

Value

`All.index.by.layer`
: Values of indices for each layer

`Best.nc`
: Best number of clusters proposed by each index and the corresponding index value.

`Best.partition`
: Partition that corresponds to the best number of clusters

Author(s)

Sarra Chair and Malika Charrad

References

Charrad M., Ghazzali N., Boiteau V., Niknafs A. (2014). "NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set.", "Journal of Statistical Software, 61(6), 1-36.", "URL http://www.jstatsoft.org/v61/i06/".

Khanchouch, I., Charrad, M., & Limam, M. (2014). A Comparative Study of Multi-SOM Algorithms for Determining the Optimal Number of Clusters. Journal of Statistical Software, 61(6), 1-36.

Examples

```
## A 4-dimensional example

set.seed(1)

data<-rbind(matrix(rnorm(100,sd=0.3),ncol=2),
         matrix(rnorm(100,mean=2,sd=0.3),ncol=2),
         matrix(rnorm(100,mean=4,sd=0.3),ncol=2),
         matrix(rnorm(100,mean=8,sd=0.3),ncol=2))

res<- multisom.batch(data,xheight= 8, xwidth= 8,"hexagonal",
                min.radius=0.00010,max.radius=0.002,
                maxit=1000,"random","gaussian","ch")

res$All.index.by.layer
res$Best.nc
res$Best.partition
```

`multisom.stochastic` *Multisom for stochastic version*

Description

This function implements the stochastic version of MultiSOM algorithm.

Usage

```
multisom.stochastic(data = NULL, xheight = 7, xwidth = 7,
                  topo = c("rectangular", "hexagonal"),
                  neighbouhood.fct =c("bubble","gaussian"),
                  dist.fcts = NULL, rlen = 100,alpha = c(0.05, 0.01),
                  radius = c(2, 1.5, 1.2, 1), index = "all")
```

Arguments

`data`	the data matrix of observations
`xheight`	the x-dimension of the map
`xwidth`	the y-dimension of the map
`topo`	the topology used to build the grid.The following are permitted: `"hexagonal"` `"rectangular"`
`neighbouhood.fct`	the neighbouhood function type. The following are permitted: `"gaussian"` `"bubble"`
`dist.fcts`	The metric used to determine the distance function. Possible choices are: `"sumofsquares"` `"euclidean"` `"manhattan"` `"tanimoto"`

rlen	the maximum number of iterations to be done
alpha	learning rate, a vector of two numbers indicating the amount of change. Default is to decline linearly from 0.05 to 0.01 over rlen updates.
radius	the radius of the neighbourhood, either given as a single number or a vector (start, stop). If it is given as a single number the radius will run from the given number to the negative value of that number; as soon as the neighbourhood gets smaller than one only the winning unit will be updated.
index	vector of the index to be calculated. This should be one of : "db", "dunn", "silhouette", "ptbiserial", "ch", "cindex", "ratkowsky", "mcclain", "gamma", "gplus", "tau", "ccc", "scott", "marriot", "trcovw", "tracew", "friedman", "rubin", "ball", "sdbw", "dindex", "hubert", "sv", "xie-beni", "hartigan", "ssi", "xu", "rayturi", "pbm", "banfeld", "all" (all indices will be used)

Value

All.index.by.layer	Values of indices for each layer.
Best.nc	Best number of clusters proposed by each index and the corresponding index value.
Best.partition	Partition that corresponds to the best number of clusters

Author(s)

Sarra Chair and Malika Charrad

Examples

```
## A real data example

data<-as.matrix(iris[,-c(5)])

res<-multisom.stochastic(data, xheight = 8, xwidth = 8,"hexagonal","gaussian",
                    dist.fcts = NULL, rlen = 100,alpha = c(0.05, 0.01),
                    radius = c(2, 1.5, 1.2, 1),c("db","ratkowsky","dunn"))

res$All.index.by.layer
res$Best.nc
```

Index