Package 'WCluster'

November 17, 2023

Type Package

Title Clustering and PCA with Weights, and Data Nuggets Clustering

Version 1.2.0

Date 2023-11-15

Author Yajie Duan [aut, cre],
Javier Cabrera [aut],
Ge Cheng [aut]

Maintainer Yajie Duan <yajieritaduan@gmail.com>

Description K-means clustering, hierarchical clustering, and PCA with observational weights and/or variable weights. It also includes the corresponding functions for data nuggets which serve as representative samples of large datasets. Cherasia et al., (2022) <doi:10.1007/978-3-031-22687-8_20>. Amaratunga et al., (2009) <doi:10.1002/9780470317129>.

Depends R (>= 3.5.0), stats, datanugget(>= 1.2.2)

Imports cluster

License GPL-2

Encoding UTF-8

NeedsCompilation no

Repository CRAN

Date/Publication 2023-11-17 17:50:02 UTC

R topics documented:

`WCluster-package` *Clustering and PCA with Observational Weights and/or Variable Weights, and Data Nuggets Clustering*

Description

This package contains functions for K-means clustering, hierarchical clustering, and PCA with observational weights and/or variable weights. It also includes the corresponding functions for data nuggets which serve as representative samples of large datasets.

Author(s)

Yajie Duan, Javier Cabrera, Ge Cheng

References

Amaratunga, D., & Cabrera, J. (2009). Exploration and analysis of DNA microarray and protein array data. *John Wiley & Sons* (Vol. 605).

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

`datanugget-package`, `Wkmeans`, `Whclust`, `Wpca`, `DN.Wkmeans`

`cluster.predict` *Predict the closest clusters for a new dataset.*

Description

Given observations with weights and cluster assignments, this function returns the cluster assignments for a new dataset by choosing the closest clusters.

Usage

```
cluster.predict(x,w = rep(1,nrow(x)),cl,newx)
```

Arguments

`x`	A data matrix (data frame, data table, matrix, etc.) containing only entries of class numeric.
`w`	Vector of length nrow(x) of weights for each observation in the dataset. Must be of class numeric or integer. If NULL, the default value is a vector of 1 with length nrow(x), i.e., weights equal 1 for all observations.
`cl`	Vector of length nrow(x) of cluster assignments for each observation in the dataset, indicating the cluster to which each observation is allocated. Must be of class integer.
`newx`	A new dataset (a data.frame), with the same variables as the learning dataset. Must be of class data.frame.

Details

To obtain the cluster assignments for a new dataset, the weighted cluster centers are calculated firstly based on observations with weights and known cluster assginments. Then, the cluster with the minimal Euclidean distance between new observation and weighted cluster center is chosen as the closest cluster. In this way, the cluster assignments for all the new observations are returned.

Value

Vector of length nrow(newx) containing the cluster assignments for each observation in the new dataset.

Author(s)

Yajie Duan, Javier Cabrera, Ge Cheng

References

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

Wkmeans

Examples

```
require(cluster)

# The Ruspini data set from the package "cluster""
data = as.matrix(ruspini)

#take the first 70 observations for clustering,
#and the last 5 observations for prediction
x = data[1:70,]
test.x = data[71:75,]

# assign random weights to observations
w = sample(1:20,nrow(x),replace = TRUE)

#k-means clustering with observational weights
cl = Wkmeans(dataset = x, k = 4, obs.weights = w, num.init = 3)

#predict the cluster assignments for the test data
cluster.predict(x,w, cl = cl$`Cluster Assignments`,newx = as.data.frame(test.x))
```

`distw` *Distance between clusters based on Ward's method for observations with weights*

Description

This function calculates distances between pairs of clusters based on Ward's method for observations with weights. Specifically, for each pair of clusters, it computes the increase of weighted sum of squares after merging them.

Usage

```
distw(x,cl,w)
```

Arguments

x	A data matrix (data frame, data table, matrix, etc.) containing only entries of class numeric.
cl	Vector of length nrow(x) of cluster assignments for each observation in the dataset, indicating the cluster to which each observation is allocated. Must be of class integer.
w	Vector of length nrow(x) of weights for each observation in the dataset. Must be of class numeric or integer. If NULL, the default value is a vector of 1 with length nrow(x), i.e., weights equal 1 for all observations.

Details

Based on the Ward method, the distance between two clusters A and B, is the increase of sum of squares after merging them, which is the merging cost of combining two clusters. Specifically, `dist(A,B) = SS(A+B) - SS(A) - SS(B)`, where SS(A+B) is sum of squares of residuals with respect to mean considering A and B as one cluster, SS(A) and SS(B) are for the cluster A and B seperately.

Here this function computes the merging costs for each pair of clusters, especially for a data set with observational weights. The sums of squares are calculated with observational weights. The distances of pairs of clusters could be used for agglomerative hierarchical clustering. The pair of clusters with minimal distance could be merged at the next step.

Value

A k by k matrix where k is the number of clusters. The lower triangular part of the matrix contains distances for pairs of clusters based on Ward's method. There are NAs on all the other positions.

Author(s)

Javier Cabrera, Yajie Duan, Ge Cheng

References

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301), 236-244.

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

Whclust

Examples

```
library(cluster)
# The Ruspini data set from the package "cluster""
x = as.matrix(ruspini)

# assign random weights to observations
w = sample(1:10,nrow(x),replace = TRUE)

# assign random clusters to observations
cl = sample(1:3,nrow(x),replace = TRUE)

#output distances between clusters based on Ward's method under the random cluster assignments
distw(x, cl, w)
```

DN.Whclust *Hierarchical Clustering for data nuggets*

Description

This function produces the hierarchical tree of data nuggets for an object of class datanugget, by agglomerative hierarchical clustering based on Ward's method considering data nugget centers and weights.

Usage

```
DN.Whclust(datanugget)
```

Arguments

datanugget	An object of class datanugget, i.e., the output of functions create.DN or refine.DN in the package datanugget.

Details

Data nuggets are a representative sample meant to summarize Big Data by reducing a large dataset to a much smaller dataset by eliminating redundant points while also preserving the peripheries of the dataset. Each data nugget is defined by a center (location), weight (importance), and scale (internal variability). Data nuggets for a large dataset could be created and refined by functions create.DN or refine.DN in the package datanugget.

Based on data nugget centers and weights, this function produces the hierarchical tree of data nuggets for an object of class datanugget, by agglomerative hierarchical clustering for data nugget centers with nugget weights as observational weights. Ward's method is used by computing the merging costs for each pair of clusters, i.e., the increase of sum of squares after merging two clusters. During the process of agglomerative hierarchical clustering, the sums of squares are calculated with data nugget weights, and the pair of clusters with minimal distance is merged at each step.

Value

An object of class *hclust* which describes the tree produced by the clustering process for data nuggets. It's the same class of object as outputs from function `hclust` in the package `stats`. See details in `hclust`. There are `print`, `plot`, and `cutree` methods for `hclust` objects.

Author(s)

Yajie Duan, Javier Cabrera, Ge Cheng

References

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301), 236-244.

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

`datanugget-package`, `create.DN`, `refine.DN`, `Whclust`, `hclust`, `distw`

Examples

```
require(datanugget)

#2-d small example with visualization
X = rbind.data.frame(matrix(rnorm(10^3, sd = 0.3), ncol = 2),
          matrix(rnorm(10^3, mean = 1, sd = 0.3), ncol = 2))


#create data nuggets
my.DN = create.DN(x = X,
                  R = 100,
                  delete.percent = .1,
                  DN.num1 = 100,
                  DN.num2 = 50,
                  no.cores = 0,
                  make.pbs = FALSE)


#refine data nuggets
my.DN2 = refine.DN(x = X,
                   DN = my.DN,
                   EV.tol = .9,
                   min.nugget.size = 2,
                   max.splits = 5,
```

```
                    no.cores = 0,
                    make.pbs = FALSE)

#plot raw dataset
plot(X)


#transform weights to get colors for plot
w_trans = my.DN2$`Data Nuggets`[, "Weight"]/sum(my.DN2$`Data Nuggets`[, "Weight"])
w_trans = w_trans/quantile(w_trans,0.8)
col = sapply(w_trans, function(t){rgb(0,min(t,1),0)})

#plot refined data nugget centers with weights
#lighter green means more weights
plot(my.DN2$`Data Nuggets`[, c("Center1",
                               "Center2")],col=col,lty = 2,pch=16, cex=0.5)


#Hierarchical Clustering for data nuggets
DN.h = DN.Whclust(my.DN2)

#print the hclust object
print(DN.h)

#plot the hierarchical tree
plot(DN.h)

#cut the hierarchical tree to get 2 clusters
k2 = cutree(DN.h,2)
table(k2)

#plot the clustering result for data nuggets
plot(my.DN2$`Data Nuggets`[, c("Center1",
                               "Center2")], col = k2, lty = 2,pch=16, cex=0.5)
```

DN.Wkmeans *K-means Clustering for data nuggets*

Description

This function clusters data nuggets for an object of class datanugget, using K-means considering data nugget centers and weights.

Usage

```
DN.Wkmeans(datanugget,
        k,
        cl.centers = NULL,
        num.init = 1,
```

```
          max.iterations = 10,
          seed = 291102)
```

Arguments

datanugget	An object of class datanugget, i.e., the output of functions create.DN or refine.DN in the package datanugget.
k	Number of desired clusters. Must be of class numeric or integer.
cl.centers	Chosen cluster centers. If NULL (default), random partition initialization would be used. If not NULL, must be a matrix containing only entries of class numeric with dimensions k by the dimension of data nugget centers.
num.init	Number of initial clusters to attempt. Ignored if cl.centers is not NULL. Must be of class numeric or integer.
max.iterations	Maximum number of iterations attempted for convergence before quitting. Must be of class numeric or integer.
seed	Random seed for replication. Must be of class numeric or integer.

Details

Data nuggets are a representative sample meant to summarize Big Data by reducing a large dataset to a much smaller dataset by eliminating redundant points while also preserving the peripheries of the dataset. Each data nugget is defined by a center (location), weight (importance), and scale (internal variability). Data nuggets for a large dataset could be created and refined by functions create.DN or refine.DN in the package datanugget.

K-means clustering with observation weigths can be used as an unsupervised learning technique to cluster observations contained in datasets that also have a measure of importance (e.g. weight) associated with them. In the case of data nuggets, this is the weight parameter associated with the data nuggets, so the centers of data nuggets are clustered using their weight parameters. The objective of the algorithm which performs this method of clustering is to minimize the weighted within cluster sum of squares (WWCSS) considering data nugget weights.

In this function, if no chosen initial cluster centers for data nuggets, random partition initialization with nugget weights is used. Each data nugget is first randomly assigned to a random cluster ID, and then the weighted cluster centers are calculated considering nugget weights. The initial cluster assignments are obtained by choosing the clusters with minimal weighted sum of squares of residuals with respect to the weighted centers.

Value

A list containing the following components:

`Cluster Assignments for data nuggets`
: Vector of length nrow(datanugget$‘Data Nuggets‘), i.e., the number of data nuggets. It contains the cluster assignments for each data nugget.

`Cluster Centers`
: k by dimension of data nuggets matrix containing the weighted cluster centers for each cluster.

`Weighted WCSS` List containing the individual WWCSS for each cluster and the combined sum of all individual WWCSS's.

`Cluster Assignments for original dataset`
Vector of length(datanugget$'Data Nugget Assignments'), i.e., number of observations in the original large dataset. It contains the cluster assignments for each observation in the original large dataset.

Author(s)

Yajie Duan, Javier Cabrera, Ge Cheng

References

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

`datanugget-package`, `create.DN`, `refine.DN`, `Wkmeans`, `wwcss`, `wss`

Examples

```
require(datanugget)

#2-d small example with visualization
X = rbind.data.frame(matrix(rnorm(10^4, sd = 0.3), ncol = 2),
          matrix(rnorm(10^4, mean = 1, sd = 0.3), ncol = 2))


#create data nuggets
my.DN = create.DN(x = X,
                  R = 500,
                  delete.percent = .1,
                  DN.num1 = 500,
                  DN.num2 = 250,
                  no.cores = 0,
                  make.pbs = FALSE)


#refine data nuggets
my.DN2 = refine.DN(x = X,
                   DN = my.DN,
                   EV.tol = .9,
                   min.nugget.size = 2,
                   max.splits = 5,
                   no.cores = 0,
                   make.pbs = FALSE)
```

```
#plot raw large dataset
plot(X)


#transform weights to get colors for plot
w_trans = my.DN2$`Data Nuggets`[, "Weight"]/sum(my.DN2$`Data Nuggets`[, "Weight"])
w_trans = w_trans/quantile(w_trans,0.8)
col = sapply(w_trans, function(t){rgb(0,min(t,1),0)})

#plot refined data nugget centers with weights
#lighter green means more weights
plot(my.DN2$`Data Nuggets`[, c("Center1",
                               "Center2")],col=col,lty = 2,pch=16, cex=0.5)



#K-means Clustering for data nuggets
DN.clus = DN.Wkmeans(datanugget = my.DN2,
            k = 2,
            num.init = 1,
            max.iterations = 5)


DN.clus$`Cluster Centers`
DN.clus$`Weighted WCSS`


#plot the clustering result for data nuggets
plot(my.DN2$`Data Nuggets`[, c("Center1",
                               "Center2")],
    col = DN.clus$`Cluster Assignments for data nuggets`, lty = 2,pch=16, cex=0.5)
points(DN.clus$`Cluster Centers`, col = 1:2, pch = 8, cex = 5)

#plot the clustering result for raw large dataset
plot(X, col = DN.clus$`Cluster Assignments for original dataset`)
```

DN.Wpca *Weighted PCA for data nuggets*

Description

This function conducts weighted PCA on data nuggets, considering data nugget centers and weights.

Usage

```
DN.Wpca(datanugget,wcol = NULL, corr = FALSE)
```

Arguments

datanugget	An object of class datanugget, i.e., the output of functions create.DN or refine.DN in the package datanugget.
wcol	Column Weights: Vector of weights for each variable of data nuggets. Must be of class numeric or integer or table. If NULL, column weights are not considered, i.e., weights equal 1 for all columns.
corr	A logical value indicating whether to use correlation matrix. This is recommended when the column weights are not equal. The default value is FALSE.

Details

Data nuggets are a representative sample meant to summarize Big Data by reducing a large dataset to a much smaller dataset by eliminating redundant points while also preserving the peripheries of the dataset. Each data nugget is defined by a center (location), weight (importance), and scale (internal variability). Data nuggets for a large dataset could be created and refined by functions create.DN or refine.DN in the package datanugget. Based on data nugget centers and weights, this function conducts weighted PCA by eigen method for data nugget centers with nugget weights as observational weights. Variable weights could also be included and considered in this function. Correlation matrix is recommended to use when the column weights are not equal.

Value

A list containing the following components:

sdev	the standard deviations of the weighted principal components (i.e., the square roots of the eigenvalues of the weighted covariance/correlation matrix).
rotation	The matrix of the loading vectors for each of the weighted prinicipal components.
x	The weighted prinicipal components.
center, scale	the weighted centering and scaling used.
wrow, wcol	row weights and column weights used.

Author(s)

Yajie Duan, Javier Cabrera, Ge Cheng

References

Amaratunga, D., & Cabrera, J. (2009). Exploration and analysis of DNA microarray and protein array data. *John Wiley & Sons* (Vol. 605).

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

datanugget-package, create.DN, refine.DN, Wpca

Examples

```
require(datanugget)

## small example
X = cbind.data.frame(rnorm(10^3),
                     rnorm(10^3),
                     rnorm(10^3))

suppressMessages({

  my.DN = create.DN(x = X,
                    R = 500,
                    delete.percent = .1,
                    DN.num1 = 500,
                    DN.num2 = 250,
                    no.cores = 0,
                    make.pbs = FALSE)

  my.DN.PCA.info = DN.Wpca(my.DN)

})

my.DN.PCA.info$sdev
my.DN.PCA.info$rotation
my.DN.PCA.info$x
```

DNcluster.predict *Predict the closest clusters for a new dataset based on clusters of data nuggets.*

Description

Given an object of class datanugget and cluster assignments for data nuggets, this function returns the cluster assignments for new observations or new data nuggets by choosing the closest clusters.

Usage

```
DNcluster.predict(datanugget, cl, newx)
```

Arguments

datanugget	An object of class datanugget, i.e., the output of functions create.DN or refine.DN in the package datanugget.
cl	Vector of length nrow(datanugget$‘Data Nuggets‘), i.e., the number of data nuggets, containing cluster assignments for each data nugget. Must be of class integer.
newx	A new dataset (a data.frame) with same variables as the original large dataset, or a new object of class datanugget with same variables of data nuggets as the learning datanugget object.

Details

To obtain the cluster assignments for new observations or new data nuggets, the weighted cluster centers are calculated firstly based on data nugget centers and weights with their known cluster assginments. Then, the cluster with the minimal Euclidean distance between new observation or new data nugget center and weighted cluster center is chosen as the closest cluster. In this way, the cluster assignments for all the new observations or new data nuggets are returned. The weights of new data nuggets are not considered here when predicting the cluster assignments.

Value

Vector of length nrow(newx) or number of new data nuggets, containing the cluster assignments for each new observation or new data nugget.

Author(s)

Yajie Duan, Javier Cabrera, Ge Cheng

References

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

DN.Whclust, DN.Wkmeans, cluster.predict

Examples

```
require(datanugget)

#2-d small example
X = rbind.data.frame(matrix(rnorm(5*10^3, sd = 0.3), ncol = 2),
          matrix(rnorm(5*10^3, mean = 1, sd = 0.3), ncol = 2))
```

```
#create data nuggets
my.DN = create.DN(x = X,
                  R = 300,
                  delete.percent = .1,
                  DN.num1 = 300,
                  DN.num2 = 150,
                  no.cores = 0,
                  make.pbs = FALSE)


#refine data nuggets
my.DN2 = refine.DN(x = X,
                   DN = my.DN,
                   EV.tol = .9,
                   min.nugget.size = 2,
                   max.splits = 5,
                   no.cores = 0,
                   make.pbs = FALSE)


#K-means Clustering for data nuggets
DN.clus = DN.Wkmeans(datanugget = my.DN2,
            k = 2,
            num.init = 1,
            max.iterations = 5)


#new observations to predict cluster assignments
newdata = matrix(rnorm(10^2, mean = 0.5, sd = 0.3), ncol = 2)

#predict the cluster assignments for the new observations
DNcluster.predict(my.DN2,
                  cl = DN.clus$`Cluster Assignments for data nuggets`,
                  newx = as.data.frame(newdata))



#predict cluster assignments for new data nuggets from a new large dataset
newdata = rbind.data.frame(matrix(rnorm(5*10^3, sd = 0.5), ncol = 2),
          matrix(rnorm(5*10^3, mean = 1, sd = 0.5), ncol = 2))

#create data nuggets
my.DN_new = create.DN(x = newdata,
                  R = 300,
                  delete.percent = .1,
                  DN.num1 = 300,
                  DN.num2 = 150,
                  no.cores = 0,
                  make.pbs = FALSE)
```

```
#refine data nuggets
my.DN2_new = refine.DN(x = newdata,
                   DN = my.DN_new,
                   EV.tol = .9,
                   min.nugget.size = 2,
                   max.splits = 5,
                   no.cores = 0,
                   make.pbs = FALSE)

#predict the cluster assignments for the new data nuggets
DNcluster.predict(my.DN2,
                  cl = DN.clus$`Cluster Assignments for data nuggets`,
                  newx = my.DN2_new)
```

`Whclust` *Hierarchical Clustering with observational weights*

Description

This function produces the hierarchical tree for observations with weights, by agglomerative hierarchical clustering based on Ward's method considering observational weights.

Usage

```
Whclust(x,w)
```

Arguments

`x`	A data matrix (of class matrix, data.frame, or data.table) containing only entries of class numeric.
`w`	Vector of length nrow(x) of weights for each observation in the dataset. Must be of class numeric or integer. If NULL, the default value is a vector of 1 with length nrow(x), i.e., weights equal 1 for all observations.

Details

Agglomerative hierarchical clustering based on Ward's method considering observational weights are used to generate the hierarchical tree. Based on the Ward method, the distance between two clusters is the increase of sum of squares after merging them, which is the merging cost of combining two clusters. This function computes the merging costs for each pair of clusters for a data set with observational weights. During the process of agglomerative hierarchical clustering, the sums of squares are calculated with observational weights, and the pair of clusters with minimal distance is merged at each step.

Value

An object of class *hclust* which describes the tree produced by the clustering process. It's the same class of object as outputs from function `hclust` in the package `stats`. See details in `hclust`. There are `print`, `plot`, and `cutree` methods for `hclust` objects.

Author(s)

Javier Cabrera, Yajie Duan, Ge Cheng

References

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

hclust, distw

Examples

```
require(cluster)
  t1 = Sys.time()

# The Ruspini data set from the package "cluster""
x = as.matrix(ruspini)

# assign random weights to observations
w = sample(1:20,nrow(x),replace = TRUE)

# hierarchical clustering with observational weights
h = Whclust(x,w)

#print the hclust object
print(h)

#plot the hierarchical tree
plot(h)

#cut the hierarchical tree to get 4 clusters
k4 = cutree(h,4)
table(k4)

#plot the clustering result
plot(x,cex = log(w),pch = 16,col = k4)
  t2 = Sys.time()
```

Wkmeans *K-means Clustering with observational weights*

Description

This function clusters data with observational weights using K-means.

Usage

```
Wkmeans(dataset,
        k,
        cl.centers = NULL,
        obs.weights = rep(1, nrow(dataset)),
        num.init = 1,
        max.iterations = 10,
        seed = 291102)
```

Arguments

dataset	A data matrix (data frame, data table, matrix, etc) containing only entries of class numeric.
k	Number of desired clusters. Must be of class numeric or integer.
cl.centers	Chosen cluster centers. If NULL (default), random partition initialization with observational weights would be used. If not NULL, must be a k by ncol(dataset) matrix containing only entries of class numeric.
obs.weights	Vector of length nrow(dataset) of weights for each observation in the dataset. Must be of class numeric or integer or table. If NULL, the default value is a vector of 1 with length nrow(dataset), i.e., weights equal 1 for all observations.
num.init	Number of initial clusters to attempt. Ignored if cl.centers is not NULL. Must be of class numeric or integer.
max.iterations	Maximum number of iterations attempted for convergence before quitting. Must be of class numeric or integer.
seed	Random seed for replication. Must be of class numeric or integer.

Details

K-means clustering with observational weights can be used as an unsupervised learning technique to cluster observations contained in datasets that also have a measure of importance (e.g. weight) associated with them. The objective of the algorithm which performs this method of clustering is to minimize the total weighted within cluster sum of squares (WWCSS) considering observational weights.

In this function, if no chosen initial cluster centers, random partition initialization with observational weights is used. Each point in the data is first randomly assigned to a random cluster ID, and then the weighted cluster centers are calculated considering observational weights. The initial cluster assignments are obtained by choosing the clusters with minimal weighted sum of squares of residuals with respect to the weighted centers.

Value

A list containing the following components:

`Cluster Assignments`
: Vector of length nrow(dataset) containing the cluster assignment for each observation.

`Cluster Centers`
: k by ncol(dataset) matrix containing the weighted cluster centers for each cluster.

`Weighted WCSS`
: List containing the individual WWCSS for each cluster and the combined sum of all individual WWCSS's.

Author(s)

Javier Cabrera, Yajie Duan, Ge Cheng

References

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

`wss`, `wwcss`, `wmean`

Examples

```
require(graphics)

x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
       matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

# assign random weights to observations
w = sample(1:20,nrow(x),replace = TRUE)

#k-means with observational weights
cl = Wkmeans(dataset = x, k = 2, obs.weights = w, num.init = 2)

plot(x,cex = log(w),pch = 16,col = cl$`Cluster Assignments`)
points(cl$`Cluster Centers`, col = 1:2, pch = 8, cex = 5)

#individual WWCSS for each cluster and the combined sum of all individual WWCSS's
cl$`Weighted WCSS`
```

```
require(cluster)

# The Ruspini data set from the package "cluster""
x = as.matrix(ruspini)

# assign random weights to observations
w = sample(1:20,nrow(x),replace = TRUE)

#k-means with observational weights
cl = Wkmeans(dataset = x, k = 4, obs.weights = w, num.init = 3)

plot(x,cex = log(w),pch = 16,col = cl$`Cluster Assignments`)
points(cl$`Cluster Centers`, col = 1:4, pch = 8, cex = 5)

#individual WWCSS for each cluster and the combined sum of all individual WWCSS's
cl$`Weighted WCSS`
```

---

wmean *Cluster Centers for observations with weights*

Description

This function computes the weighted cluster centers for a set of cluster assignments provided to a dataset with observational weights.

Usage

```
wmean(x,cl,w)
```

Arguments

x	A data matrix (data frame, data table, matrix, etc.) containing only entries of class numeric.
cl	Vector of length nrow(x) of cluster assignments for each observation in the dataset, indicating the cluster to which each observation is allocated. Must be of class integer.
w	Vector of length nrow(x) of weights for each observation in the dataset. Must be of class numeric or integer. If NULL, the default value is a vector of 1 with length nrow(x), i.e., weights equal 1 for all observations.

Details

In this function, the function `weighted.mean` in the `stats` package is used to calculate the cluster centers for each cluster with observational weights.

Value

A matrix of cluster centres. Each column is a weighted center for one cluster.

Author(s)

Javier Cabrera, Yajie Duan, Ge Cheng

References

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

Wkmeans

Examples

```
require(cluster)
# The Ruspini data set from the package "cluster""
x = as.matrix(ruspini)

# assign random weights to observations
w = sample(1:10,nrow(x),replace = TRUE)

# assign random clusters to observations
cl = sample(1:3,nrow(x),replace = TRUE)

#output the weighted cluster centers for each cluster under the random cluster assignments
wmean(x, cl, w)
```

Wpca *Weighted PCA*

Description

This function performs PCA on the given data matrix, with row and column weights.

Usage

```
Wpca(x, wrow = rep(1, nrow(x)), wcol = rep(1, ncol(x)), corr = FALSE)
```

Arguments

x	A data matrix (data frame, data table, matrix, etc) containing only entries of class numeric.
wrow	Row Weights: vector of length nrow(x) of weights for each observation in the dataset. Must be of class numeric or integer or table. If NULL, the default value is a vector of 1 with length nrow(x), i.e., weights equal 1 for all observations.
wcol	Column Weights: Vector of length ncol(x) of weights for each variable in the dataset. Must be of class numeric or integer or table. If NULL, the default value is a vector of 1 with length ncol(x), i.e., weights equal 1 for all columns.
corr	A logical value indicating whether to use correlation matrix. This is recommended when the column weights are not equal. The default value is FALSE.

Details

PCA with row and column weights is conducted by eigen method.

Value

A list containing the following components:

sdev	the standard deviations of the weighted principal components (i.e., the square roots of the eigenvalues of the weighted covariance/correlation matrix).
rotation	The matrix of the loading vectors for each of the weighted prinicipal components.
x	The weighted prinicipal components.
center, scale	the weighted centering and scaling used.
wrow, wcol	row weights and column weights used.

Author(s)

Javier Cabrera, Yajie Duan, Ge Cheng

References

Amaratunga, D., & Cabrera, J. (2009). Exploration and analysis of DNA microarray and protein array data. *John Wiley & Sons* (Vol. 605).

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

Examples

```
require(cluster)

# The Ruspini data set from the package "cluster""
x = as.matrix(ruspini)

# assign random weights to observations
w = sample(1:20,nrow(x),replace = TRUE)

#PCA with observational weights
res = Wpca(x, wrow = w)

#weighted prinicipal components
pc = res$x
pc

#loading vectors
loadings = res$rotation
loadings
```

`wss` *Sums of squares of residuals for observations with weights*

Description

This function calculates sums of squares of residuals with respect to mean for observations with weights.

Usage

```
wss(x,w = rep(1,nrow(x)))
```

Arguments

`x`	A data matrix (data frame, data table, matrix, etc.) containing only entries of class numeric.
`w`	Vector of length nrow(x) of weights for each observation in the dataset. Must be of class numeric or integer. If NULL, the default value is a vector of 1 with length nrow(x), i.e., weights equal 1 for all observations.

Details

In this function, for a dataset with observational weights, the weighted mean for the dataset is calculated first. Based on it, the weighted sum of squares of residuals with respect to the weighted mean is calculated with observational weights. This could be used to calculate weighted within-cluster sum of squares for one cluster of data with observational weights.

Value

a length-one numeric vector.

Author(s)

Javier Cabrera, Yajie Duan, Ge Cheng

Examples

```
require(cluster)
# The Ruspini data set from the package "cluster""
x = as.matrix(ruspini)

# assign random weights to observations
w = sample(1:10,nrow(x),replace = TRUE)

wss(x,w)
```

---

wwcss *Weighted Within Cluster Sum of Squares*

Description

This function computes the weighted within cluster sum of squares (WWCSS) for a set of cluster assignments provided to a dataset with observational weights.

Usage

```
wwcss(x, cl, w = rep(1,length(x)), groupSum = FALSE)
```

Arguments

x	A data matrix (data frame, data table, matrix, etc.) containing only entries of class numeric.
cl	Vector of length nrow(x) of cluster assignments for each observation in the dataset, indicating the cluster to which each observation is allocated. Must be of class integer.
w	Vector of length nrow(x) of weights for each observation in the dataset. Must be of class numeric or integer. If NULL, the default value is a vector of 1 with length nrow(x), i.e., weights equal 1 for all observations.
groupSum	A logical value indicating whether the weighted within-cluster sum of squres (WWCSS) of each cluster should be returned. If TRUE the total WWCSS and WWCSS for each cluster are returned. If FALSE (the default) only the total WWCSS is returned.

Details

This function is used to evaluate clustering results for observations with weights, and also used for optimizing the cluster assignments in the Wkmeans function.

Value

A list containing the following components:

`WWCSS`	If requested by `groupSum`, vector of individual WWCSS's for each cluster
`TotalWWCSS`	Combined sum of all individual WWCSS's.

Author(s)

Javier Cabrera, Yajie Duan, Ge Cheng

References

Cherasia, K. E., Cabrera, J., Fernholz, L. T., & Fernholz, R. (2022). Data Nuggets in Supervised Learning. *In Robust and Multivariate Statistical Methods: Festschrift in Honor of David E. Tyler* (pp. 429-449). Cham: Springer International Publishing.

Beavers, T., Cheng, G., Duan, Y., Cabrera, J., Lubomirski, M., Amaratunga, D., Teigler, J. (2023). Data Nuggets: A Method for Reducing Big Data While Preserving Data Structure (Submitted for Publication)

See Also

`Wkmeans`

Examples

```
require(cluster)
# The Ruspini data set from the package "cluster""
x = as.matrix(ruspini)

# assign random weights to observations
w = sample(1:10,nrow(x),replace = TRUE)

# assign random clusters to observations
cl = sample(1:3,nrow(x),replace = TRUE)

#output the total WWCSS and WWCSS for each cluster for the cluster assignments
wwcss(x, cl, w, groupSum = TRUE)
```

Index