Package ‘varclus’

June 26, 2019

Type Package
Title Variables Clustering
Version 0.9.4
Date 2019-06-08
Author Piotr Sobczyk, Stanislaw Wilczynski, Julie Josse, Malgorzata Bogdan
Maintainer Piotr Sobczyk <pj.sobczyk@gmail.com>
Description Performs clustering of quantitative variables, assuming that clusters lie in low-dimensional subspaces. Segmentation of variables, number of clusters and their dimensions are selected based on BIC. Candidate models are identified based on many runs of K-means algorithm with different random initializations of cluster centers.
Encoding UTF-8
License GPL-3
Depends R (>= 3.2.1)
Imports RcppEigen, foreach, parallel, doParallel, doRNG, pesel
Suggests knitr, mclust, rmarkdown, testthat
NeedsCompilation no
VignetteBuilder knitr
RoxygenNote 6.1.1
Repository CRAN
Date/Publication 2019-06-26 10:10:37 UTC

R topics documented:

data.simulation .................................................. 2
data.simulation.factors ........................................ 3
integration ........................................................ 4
misclassification ................................................... 5
mlcc.bic ............................................................ 6
mlcc.kmeans ......................................................... 7
data.simulation

Simulates subspace clustering data

Description

Generates data for simulation with a low-rank subspace structure: variables are clustered and each cluster has a low-rank representation. Factors than span subspaces are not shared between clusters.

Usage

data.simulation(n = 100, SNR = 1, K = 10, numb.vars = 30, max.dim = 2, min.dim = 1, equal.dims = TRUE)

Arguments

n
An integer, number of individuals.

SNR
A numeric, signal to noise ratio measured as variance of the variable, element of a subspace, to the variance of noise.

K
An integer, number of subspaces.

numb.vars
An integer, number of variables in each subspace.

max.dim
An integer, if equal.dims is TRUE then max.dim is dimension of each subspace. If equal.dims is FALSE then subspaces dimensions are drawn from uniform distribution on [min.dim,max.dim].

min.dim
An integer, minimal dimension of subspace.

equal.dims
A boolean, if TRUE (value set by default) all clusters are of the same dimension.

Value

A list consisting of:

X
matrix, generated data

signals
matrix, data without noise

dims
vector, dimensions of subspaces

factors
matrix, columns of which span subspaces

s
vector, true partition of variables

Examples

sim.data <- data.simulation()
sim.data2 <- data.simulation(n = 30, SNR = 2, K = 5, numb.vars = 20, max.dim = 3, equal.dims = FALSE)
**data.simulation.factors**

*Simulates subspace clustering data with shared factors*

### Description

Generating data for simulation with a low-rank subspace structure: variables are clustered and each cluster has a low-rank representation. Factors that span subspaces are shared between clusters.

### Usage

```r
data.simulation.factors(n = 100L, SNR = 1L, K = 10L, numb.vars = 30L,
                        numb.factors = 10L, min.dim = 1L, max.dim = 2L, equal.dims = TRUE,
                        separation.parameter = 0.1L)
```

### Arguments

- **n**: An integer, number of individuals.
- **SNR**: A numeric, signal to noise ratio measured as variance of the variable, element of a subspace, to the variance of noise.
- **K**: An integer, number of subspaces.
- **numb.vars**: An integer, number of variables in each subspace.
- **numb.factors**: An integer, number of factors from which subspaces basis will be drawn.
- **min.dim**: An integer, minimal dimension of subspace.
- **max.dim**: An integer, if equal.dims is TRUE then max.dim is dimension of each subspace. If equal.dims is FALSE then subspaces dimensions are drawn from uniform distribution on [min.dim, max.dim].
- **equal.dims**: A boolean, if TRUE (value set by default) all clusters are of the same dimension.
- **separation.parameter**: A numeric, coefficients of variables in each subspace basis are drawn from range [separation.parameter, 1]

### Value

A list consisting of:

- **X**: matrix, generated data
- **signals**: matrix, data without noise
- **factors**: matrix, columns of which span subspaces
- **indices**: list of vectors, indices of factors that span subspaces
- **dims**: vector, dimensions of subspaces
- **s**: vector, true partition of variables
integration

Computes integration and acontamination of the clustering

Description

Integration and acontamination are measures of the quality of a clustering with a reference to a true partition. Let $X = (x_1, \ldots, x_p)$ be the data set, $A$ be a partition into clusters $A_1, \ldots, A_n$ (true partition) and $B$ be a partition into clusters $B_1, \ldots, B_m$. Then for cluster $A_j$ integration is equal to:

$$\text{Int}(A_j) = \frac{\max_{k=1, \ldots, m} \# \{i \in \{1, \ldots, p\} : x_i \in A_j \land x_i \in B_k\}}{\# A_j}$$

The $B_k$ for which the value is maximized is called the integrating cluster of $A_j$. Then the integration for the whole clustering equals is $\text{Int}(A, B) = \frac{1}{n} \sum_{j=1}^{n} \text{Int}(A_j)$. The acontamination is defined by:

$$\text{Acont}(A_j) = \frac{\# \{i \in \{1, \ldots, p\} : x_i \in A_j \land x_i \in B_k\}}{\# B_k}$$

where $B_k$ is the integrating cluster for $A_j$. Then the acontamination for the whole dataset is $\text{Acont}(A, B) = \frac{1}{n} \sum_{j=1}^{n} \text{Acont}(A_j)$

Usage

integration(group, true_group)

Arguments

- group: A vector, first partition.
- true_group: A vector, second (reference) partition.

Value

An array containing values of integration and acontamination.

References


Examples

```r
sim.data <- data.simulation()
sim.data2 <- data.simulation(n = 30, SNR = 2, K = 5, numb.vars = 20,
                           numb.factors = 10, max.dim = 3, equaldims = FALSE, separation.parameter = 0.2)

integration(sim.data$segmentation, true_segmentation)
```
Description

Missclassification is a commonly used performance measure in subspace clustering. It allows to compare two partitions with the same number of clusters.

Usage

misclassification(group, true_group, M, K)

Arguments

group A vector, first partition.
true_group A vector, second (reference) partition.
M An integer, maximal number of elements in one class.
K An integer, number of classes.

Details

As getting exact value of misclassification requires checking all permutations and is therefore intractable even for modest number of clusters, a heuristic approach is proposed. It is assumed that there are K classes of maximum M elements. Additional requirement is that classes labels are from range [1, K].

Value

Misclassification rate.

References


Examples

```r
sim.data <- data.simulation(n = 100, SNR = 1, K = 5, numb.vars = 30, max.dim = 2)
mlcc.fit <- mlcc.reps(sim.data$x, numb.clusters = 5, numb.runs = 20, max.dim = 2, numb.cores=1)
misclassification(mlcc.fit$segmentation, sim.data$s, 30, 5)

# one can use this function not only for clusters
partition1 <- sample(10, 300, replace = TRUE)
partition2 <- sample(10, 300, replace = TRUE)
misclassification(partition1, partition1, max(table(partition1)), 10)
misclassification(partition1, partition2, max(table(partition2)), 10)
```
Multiple Latent Components Clustering - Subspace clustering with automatic estimation of number of clusters and their dimension

Description
This function is an implementation of Multiple Latent Components Clustering (MLCC) algorithm which clusters quantitative variables into a number, chosen using mBIC, of groups. For each considered number of clusters in `numb.clusters`, `mlcc.reps` function is called. It invokes K-means based algorithm (`mlcc.kmeans`) finding local minimum of mBIC, which is run a given number of times (`numb.runs`) with different initializations. The best partition is chosen with mBIC (see `mlcc.reps` function).

Usage
```
mlccNbic(x, numbNclusters = 1:10, numbNruns = 30, stopNcriterion = 1,
maxNiter = 30, maxNdim = 4, scale = TRUE, numbNcores = NULL,
        greedy = TRUE, estimateNdimensions = TRUE, verbose = FALSE,
        flatNprior = FALSE, showNwarnings = FALSE)
```

Arguments
- **x**: A data frame or a matrix with only continuous variables.
- **numbNclusters**: A vector, numbers of clusters to be checked.
- **numbNruns**: An integer, number of runs (initializations) of `mlcc.kmeans`.
- **stopNcriterion**: An integer, if an iteration of `mlcc.kmeans` algorithm makes less changes in partitions than `stop.criterion`, `mlcc.kmeans` stops.
- **maxNiter**: An integer, maximum number of iterations of the loop in `mlcc.kmeans` algorithm.
- **maxNdim**: An integer, if `estimate.dimensions` is FALSE then `max.dim` is dimension of each subspace. If `estimate.dimensions` is TRUE then subspaces dimensions are estimated from the range [1, `max.dim`].
- **scale**: A boolean, if TRUE (value set by default) then variables in dataset are scaled to zero mean and unit variance.
- **numbNcores**: An integer, number of cores to be used, by default all cores are used.
- **greedy**: A boolean, if TRUE (value set by default) the clusters are estimated in a greedy way - first local minimum of mBIC is chosen.
- **estimateNdimensions**: A boolean, if TRUE (value set by default) subspaces dimensions are estimated.
- **verbose**: A boolean, if TRUE plot with mBIC values for different numbers of clusters is produced and values of mBIC, computed for every number of clusters and subspaces dimensions, are printed (value set by default is FALSE).
- **flatNprior**: A boolean, if TRUE then, instead of an informative prior that takes into account number of models for a given number of clusters, flat prior is used.
- **showNwarnings**: A boolean, if set to TRUE all warnings are displayed, default value is FALSE.
Value
An object of class mlcc.fit consisting of

- segmentation: a vector containing the partition of the variables
- BIC: numeric, value of mBIC
- subspacesDimensions: a list containing dimensions of the subspaces
- nClusters: an integer, estimated number of clusters
- factors: a list of matrices, basis for each subspace
- all.fit: a list of segmentation, mBIC, subspaces dimension for all numbers of clusters considered for an estimated subspace dimensions
- all.fit.dims: a list of lists of segmentation, mBIC, subspaces dimension for all numbers of clusters and subspaces dimensions considered

Examples

```r
sim.data <- data.simulation(n = 50, SNR = 1, K = 3, numb.vars = 50, max.dim = 3)
mlcc.res <- mlcc.bic(sim.data$X, numb.clusters = 1:5, numb.runs = 20, numb.cores = 1, verbose=TRUE)
show.clusters(sim.data$X, mlcc.res$segmentation)
```

mlcc.kmeans Multiple Latent Components Clustering - kmeans algorithm

Description
Performs k-means based subspace clustering. Center of each cluster is some number of principal components. This number can be fixed or estimated by PESEL. Similarity measure between variable and a cluster is calculated using BIC.

Usage

```r
mlcc.kmeans(X, number.clusters = 2, stop.criterion = 1,
    max.iter = 30, max.subspace.dim = 4, initial.segmentation = NULL,
    estimate.dimensions = TRUE, show.warnings = FALSE)
```

Arguments

- `X`: A matrix with only continuous variables.
- `number.clusters`: An integer, number of clusters to be used.
- `stop.criterion`: An integer indicating how many changes in partitions triggers stopping the algorithm.
- `max.iter`: An integer, maximum number of iterations of k-means loop.
**max.subspace.dim**

An integer, maximum dimension of subspaces.

**initial.segmentation**

A vector, initial segmentation of variables to clusters.

**estimate.dimensions**

A boolean, if TRUE (value set by default) subspaces dimensions are estimated.

**show.warnings**

A boolean, if set to TRUE all warnings are displayed, default value is FALSE.

**Value**

A list consisting of:

- **segmentation**  a vector containing the partition of the variables
- **pcas**          a list of matrices, basis vectors for each cluster (subspace)

**References**

*Bayesian dimensionality reduction with PCA using penalized semi-integrated likelihood*, Piotr Sobczyk, Malgorzata Bogdan, Julie Josse

**Examples**

```r
sim.data <- data.simulation(n = 50, SNR = 1, K = 5, numb.vars = 50, max.dim = 3)
mlcc.res <- mlcc.kmeans(sim.data$X, number.clusters = 5, max.iter = 20, max.subspace.dim = 3)
show.clusters(sim.data$X, mlcc.res$segmentation)
```

**mlcc.reps**

*Multiple Latent Components Clustering - Subspace clustering assuming that the number of clusters is known*

**Description**

For a fixed number of cluster function returns the best partition and basis for each subspace.

**Usage**

```r
mlcc.reps(X, numb.clusters = 2, numb.runs = 30, stop.criterion = 1, 
max.iter = 30, initial.segmentations = NULL, max.dim = 4, 
scale = TRUE, numb.cores = NULL, estimate.dimensions = TRUE, 
flat.prior = FALSE, show.warnings = FALSE)
```
Arguments

- **X**  
  A data frame or a matrix with only continuous variables.

- **numb.clusters**  
  An integer, number of cluster.

- **numb.runs**  
  An integer, number of runs of `mlcc.kmeans` algorithm with random initialization.

- **stop.criterion**  
  An integer, if an iteration of `mlcc.kmeans` algorithm makes less changes in partitions than `stop.criterion`, `mlcc.kmeans` stops.

- **max.iter**  
  max.iter An integer, maximum number of iterations of the loop in `mlcc.kmeans` algorithm.

- **initial.segmentations**  
  A list of vectors, segmentations that user wants to be used as an initial segmentation in `mlcc.kmeans` algorithm.

- **max.dim**  
  An integer, maximal dimension of subspaces.

- **scale**  
  A boolean, if TRUE (value set by default) then variables in dataset are scaled to zero mean and unit variance.

- **numb.cores**  
  An integer, number of cores to be used, by default all cores are used.

- **estimate.dimensions**  
  A boolean, if TRUE (value set by default) subspaces dimensions are estimated.

- **flat.prior**  
  A boolean, if TRUE then, instead of a prior that takes into account number of models for a given number of clusters, flat prior is used.

- **show.warnings**  
  A boolean, if set to TRUE all warnings are displayed, default value is FALSE.

Details

In more detail, an algorithm `mlcc.kmeans` is run a `numb.runs` of times with random or custom initializations. The best partition is selected according to the BIC.

Value

A list consisting of

- **segmentation**  
  a vector containing the partition of the variables

- **BIC**  
  a numeric, value of the mBIC

- **basis**  
  a list of matrices, the factors for each of the subspaces

Examples

```r
sim.data <- data.simulation(n = 50, SNR = 1, K = 5, numb.vars = 50, max.dim = 3)
mlcc.res <- mlcc.reps(sim.data$X, numb.clusters = 5, numb.runs = 20, max.dim = 4, numb.cores = 1)
show.clusters(sim.data$X, mlcc.res$segmentation)
```
**show.clusters**

*Print clusters obtained from MLCC*

**Description**

Print clusters obtained from MLCC

**Usage**

```r
show.clusters(data, segmentation)
```

**Arguments**

- `data`: The original data set.
- `segmentation`: A vector, segmentation of variables into clusters.

---

**varclust**

*Variable Clustering with Multiple Latent Components Clustering algorithm*

**Description**

Package varclust performs clustering of variables, according to a probabilistic model, which assumes that each cluster lies in a low dimensional subspace. Segmentation of variables, number of clusters and their dimensions are selected based on the appropriate implementation of the Bayesian Information Criterion.

**Details**

The best candidate models are identified by the specific implementation of K-means algorithm, in which cluster centers are represented by some number of orthogonal factors (principal components of the variables within a cluster) and similarity between a given variable and a cluster center depends on residuals from a linear model fit. Based on the Bayesian Information Criterion (BIC), sums of squares of residuals are appropriately scaled, which allows to avoid an over-excessive attraction by clusters with larger dimensions. To reduce the chance that the local minimum of modified BIC (mBIC) is obtained instead of the global one, for every fixed number of clusters in a given range K-means algorithm is run large number of times, with different random initializations of cluster centers.

The main function of package `varclust` is `mlccNbic` which allows clustering variables in a data with unknown number of clusters. Variable partition is computed with k-means based algorithm. Number of clusters and their dimensions are estimated using mBIC and PESEL respectively. If the number of clusters is known one might use function `mlcc.reps`, which takes number of clusters as a parameter. For `mlcc.reps` one might specify as well some initial segmentation for k-means algorithm. This can be useful if user has some a priori knowledge about clustering.
We provide also two functions to simulate datasets with described structure. The function `data.simulation` generates the data so that the subspaces are independent and `data.simulation.factors` generates the data where some factors are shared between the subspaces.

We also provide function measures of quality of clustering. `misclassification` computes misclassification rate between two partitions. This performance measure is extensively used in image segmentation. The other measure is implemented as `integration` function.

Version: 0.9.4

Author(s)

Piotr Sobczyk, Stanislaw Wilczynski, Julie Josse, Malgorzata Bogdan

Maintainer: Piotr Sobczyk <pj.sobczyk@gmail.com>

Examples

```r
sim.data <- data.simulation(n = 50, SNR = 1, K = 3, numb.vars = 50, max.dim = 3)
mlcc.bic(sim.data$x, numb.clusters = 1:5, numb.runs = 20, numb.cores = 1, verbose = TRUE)
mlcc.reps(sim.data$x, numb.clusters = 3, numb.runs = 20, numb.cores = 1)
```
Index

data.simulation, 2, 11
data.simulation.factors, 3, 11

integration, 4, 11

misclassification, 5, 11
mlcc.bic, 6, 10
mlcc.kmeans, 6, 7, 9
mlcc.reps, 6, 8, 10

show.clusters, 10

varclust, 10
varclust-package (varclust), 10