

Package ‘ordinalClust’

June 19, 2019

Type Package

Title Ordinal Data Clustering, Co-Clustering and Classification

Version 1.3.3

Date 2019-06-19

Author Margot Selosse, Julien Jacques, Christophe Biernacki

Maintainer Margot Selosse <margot.selosse@gmail.com>

Description Ordinal data classification, clustering and co-clustering using model-based approach with the Bos distribution for ordinal data (Christophe Biernacki and Julien Jacques (2016) <doi:10.1007/s11222-015-9585-2>).

License GPL (>= 2)

Imports Rcpp (>= 0.12.11), methods

LinkingTo Rcpp, RcppProgress, RcppArmadillo, BH

Suggests knitr, rmarkdown, caret, ggplot2

VignetteBuilder knitr

LazyData true

SystemRequirements C++11

Depends R (>= 3.3)

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-06-19 17:20:03 UTC

R topics documented:

bosclassif	2
bosclust	4
boscoclust	6
dataqol	9
dataqol.classif	9
Msimulated	10
pejSim	10

plot	11
predict	11
summary	12

Index	13
--------------	-----------

bosclassif	<i>Function to perform a classification</i>
------------	---

Description

This function performs a classification on a dataset with features of the ordinal kind, and a label variable of the integer type (1,2,...,kr). The classification function proposes two classification models. The first one, (chosen by the argument `kc=0`), is a multivariate BOS model assuming that, conditionally on the class of the observations, the feature are independent. The second model is a parsimonious version of the first model. Parsimony is introduced by grouping the features into clusters (as in co-clustering) and assuming that the features of a cluster have a common distribution. If the data contains ordinal features with D different numbers of levels, the data is going to be seen as D matrices of ordinal data.

Usage

```
bosclassif(x, y, idx_list=c(1), kr, kc=0, init, nbSEM, nbSEMBurn,
           nbindmini, m=0, percentRandomB=0)
```

Arguments

<code>x</code>	Matrix made of ordinal data, of dimension $N \times J_{tot}$. The features with same numbers of levels must be placed side by side. The missing values should be coded as NA.
<code>y</code>	Vector of length N . It should represent the classes corresponding to each row of <code>x</code> . Must be labelled with integers (1,2,...,kr).
<code>idx_list</code>	Vector of length D . This argument is useful when variables have different numbers of levels. Element d should indicate where the variables with number of levels $m[d]$ begins in matrix <code>x</code> .
<code>kr</code>	Number of row classes.
<code>kc</code>	Vector of length D . d^{th} element indicates the number of column clusters. Set to 0 to choose a classical multivariate BOS model.
<code>m</code>	Vector of length D . d^{th} element defines the ordinal data's number of levels.
<code>nbSEM</code>	Number of SEM-Gibbs iterations realized to estimate parameters.
<code>nbSEMBurn</code>	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to <code>nbSEM</code> .
<code>nbindmini</code>	Minimum number of cells belonging to a block.
<code>init</code>	String that indicates the kind of initialisation. Must be one of the following words : "kmeans", "random" or "randomBurnin".
<code>percentRandomB</code>	Vector of length 1. Indicates the percentage of resampling when <code>init</code> is equal to "randomBurnin".

Value

Return an object. The slots are:

@zr	Vector of length N with resulting row partitions.
@zc	List of length D. d th item is a vector of length J[d] representing the columns partitions for the group of variables d.
@J	Vector of length D. d th item represents the number of columns for d th group of variables.
@W	List of length D. Item d is a matrix of dimension J*kc[d] such that W[j,h]=1 if j belongs to cluster h.
@V	Matrix of dimension N*kr such that V[i,g]=1 if i belongs to cluster g.
@icl	ICL value for co-clustering.
@kr	Number of row classes.
@name	Name of the result.
@number_distrib	Number of groups of variables.
@pi	Vector of length kr. Row mixing proportions.
@rho	List of length D. d th item represents the column mixing proportion for d th group of variables.
@dlist	List of length d. d th item represents the indexes of group of variables d.
@kc	Vector of length D. d th element represents the number of clusters column H for d th group of variables.
@m	Vector of length D. d th element represents the number of levels of d th group of variables.
@nbSEM	Number of SEM-Gibbs algorithm iteration.
@params	List of length D. d th item represents the blocks parameters for group of variables d.
@xhat	List of length D. d th item represents the d th group of variables dataset, with missing values completed.

Author(s)

Margot Seloisse, Julien Jacques, Christophe Biernacki.

Examples

```
# loading the real dataset
data("dataqol.classif")

set.seed(5)

# loading the ordinal data
M <- as.matrix(dataqol.classif[,2:29])
```

```

# creating the classes values
y <- as.vector(dataqol.classif$death)

# sampling datasets for training and to predict
nb.sample <- ceiling(nrow(M)*2/3)
sample.train <- sample(1:nrow(M), nb.sample, replace=FALSE)

M.train <- M[sample.train,]
M.validation <- M[-sample.train,]
nb.missing.validation <- length(which(M.validation==0))
m <- c(4)
M.validation[which(M.validation==0)] <- sample(1:m, nb.missing.validation,replace=TRUE)

y.train <- y[sample.train]
y.validation <- y[-sample.train]

# configuration for SEM algorithm
nbSEM=50
nbSEMBurn=40
nbindmini=1
init="kmeans"

# number of classes to predict
kr <- 2
# different kc to test with cross-validation
kcol <- 1

res <- bosclassif(x=M.train,y=y.train,kr=kr,kc=kcol,m=m,
                 nbSEM=nbSEM,nbSEMBurn=nbSEMBurn,
                 nbindmini=nbindmini,init=init)

predictions <- predict(res, M.validation)

```

bosclust

Function to perform a clustering

Description

This function performs a clustering on ordinal data by using the multiple latent block model (cf references for further details). It allows the user to define D groups of variables that have different number of levels. A BOS distribution is used, and the parameters inference is realized with an SEM-Gbbs algorithm.

Usage

```
bosclust(x, idx_list=c(1), kr, init, nbSEM, nbSEMBurn,
         nbindmini, m=0, percentRandomB=0)
```

Arguments

<code>x</code>	Matrix made of ordinal data, of dimension $N \times J_{tot}$. The features with same numbers of levels must be placed side by side. The missing values should be coded as NA.
<code>idx_list</code>	Vector of length D . This argument is useful when variables have different numbers of levels. Element d should indicate where the variables with number of levels $m[d]$ begins in matrix x .
<code>kr</code>	Number of row clusters.
<code>m</code>	Vector of length D . d^{th} element defines the ordinal data's number of levels.
<code>nbSEM</code>	Number of SEM-Gibbs iterations realized to estimate parameters.
<code>nbSEMBurn</code>	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to <code>nbSEM</code> .
<code>nbindmini</code>	Minimum number of cells belonging to a block.
<code>init</code>	String that indicates the kind of initialisation. Must be one of the following words : "kmeans", "random" or "randomBurnin".
<code>percentRandomB</code>	Vector of length 1. Indicates the percentage of resampling when <code>init</code> is equal to "randomBurnin".

Value

<code>@V</code>	Matrix of dimension $N \times kr$ such that $V[i,g]=1$ if i belongs to cluster g .
<code>@zr</code>	Vector of length N with resulting row partitions.
<code>@pi</code>	Vector of length kr . Row mixing proportions.
<code>@m</code>	Vector of length D . d^{th} element represents the number of levels of d^{th} group of variables.
<code>@icl</code>	ICL value for clustering.
<code>@name</code>	Name of the result.
<code>@params</code>	List of length D . d^{th} item stores the resulting position and precision parameters μ and π .
<code>@paramschain</code>	List of length <code>nbSEMBurn</code> . For each iteration of the SEM-Gibbs algorithm, the parameters of the blocks are stored.
<code>@xhat</code>	List of length D . d^{th} item represents the d^{th} group of variables dataset, with missing values completed.
<code>@zrchain</code>	Matrix of dimension <code>nbSEM</code> \times N . Row i represents the row cluster partitions at iteration i .
<code>@pichain</code>	List of length <code>nbSEM</code> . Item i is a vector of length kr which contains the row mixing proportions at iteration i .

Author(s)

Margot Seloisse, Julien Jacques, Christophe Biernacki.

Examples

```
library(ordinalClust)
data("dataqol")
set.seed(5)

# loading the ordinal data
M <- as.matrix(dataqol[,2:29])

m = 4

krow = 4

nbSEM=50
nbSEMBurn=40
nbindmini=2
init = "random"

object <- boscoclust(x=M,kr=krow, m=m, nbSEM=nbSEM,
                    nbSEMBurn=nbSEMBurn, nbindmini=nbindmini, init=init)
```

boscoclust

Function to perform a co-clustering

Description

This function performs a co-clustering on ordinal data by using the latent block model (cf references for further details). A BOS distribution is used, and the parameters inference is realized with an SEM-Gbbs algorithm.

Usage

```
boscoclust(x=matrix(0,nrow=1,ncol=1), idx_list=c(1), kr, kc, init, nbSEM, nbSEMBurn,
          nbRepeat=1, nbindmini, m=0, percentRandomB=0)
```

Arguments

x Matrix made of ordinal data, of dimension $N \times J_{tot}$. The features with same numbers of levels must be placed side by side. The missing values should be coded as NA.

<code>idx_list</code>	Vector of length D . This argument is useful when variables have different numbers of levels. Element d should indicate where the variables with number of levels $m[d]$ begins in matrix x .
<code>kr</code>	Number of row classes.
<code>kc</code>	Vector of length D . d^{th} element indicates the number of column clusters. Set to 0 to choose a classical multivariate BOS model.
<code>m</code>	Vector of length D . d^{th} element defines the ordinal data's number of levels.
<code>nbSEM</code>	Number of SEM-Gibbs iterations realized to estimate parameters.
<code>nbSEMBurn</code>	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to <code>nbSEM</code> .
<code>nbRepeat</code>	Number of times sampling on rows and on columns will be done at each SEM-Gibbs iteration.
<code>nbindmini</code>	Minimum number of cells belonging to a block.
<code>init</code>	String that indicates the kind of initialisation. Must be one of the following words: "kmeans", "random" or "randomBurnin".
<code>percentRandomB</code>	Vector of length 2. Indicates the percentage of resampling when <code>init</code> is equal to "randomBurnin".

Value

<code>@v</code>	Matrix of dimension $N \times kr$ such that $V[i,g]=1$ if i belongs to cluster g .
<code>@icl</code>	ICL value for co-clustering.
<code>@name</code>	
<code>@paramschain</code>	List of length <code>nbSEMBurn</code> . For each iteration of the SEM-Gibbs algorithm, the parameters of the blocks are stored.
<code>@pichain</code>	List of length <code>nbSEM</code> . Item i is a vector of length kr which contains the row mixing proportions at iteration i .
<code>@rhochain</code>	List of length <code>nbSEM</code> . Item i is a list of length D whose d^{th} contains the column mixing proportions of groups of variables d , at iteration i .
<code>@zc</code>	List of length D . d^{th} item is a vector of length $J[d]$ representing the columns partitions for the group of variables d .
<code>@zr</code>	Vector of length N with resulting row partitions.
<code>@w</code>	List of length D . Item d is a matrix of dimension $J \times kc[d]$ such that $W[j,h]=1$ if j belongs to cluster h .
<code>@m</code>	Vector of length D . d^{th} element represents the number of levels of d^{th} group of variables.
<code>@params</code>	List of length D . d^{th} item represents the blocks parameters for group of variables d .
<code>@pi</code>	Vector of length kr . Row mixing proportions.
<code>@rho</code>	List of length D . d^{th} item represents the column mixing proportion for d^{th} group of variables.

@xhat	List of length D. d th item represents the d th group of variables dataset, with missing values completed.
@zrchain	Matrix of dimension nbSEM*N. Row i represents the row cluster partitions at iteration i.
@zcchain	List of length D. Item d is a matrix of dimension nbSEM*J[d]. Row i represents the column cluster partitions at iteration i.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
library(ordinalClust)

# loading the real dataset
data("dataqol")
set.seed(5)

# loading the ordinal data
M <- as.matrix(dataqol[,2:29])

# defining different number of categories:
m=4

# defining number of row and column clusters
krow = 5
kcol = 4

# configuration for the inference
nbSEM=50
nbSEMBurn=40
nbindmini=2
init = "kmeans"

# Co-clustering execution
object <- boscoclust(x=M,kr=krow,kc=kcol,m=m,nbSEM=nbSEM,
                    nbSEMBurn=nbSEMBurn, nbindmini=nbindmini, init=init)
```

dataqol	<i>Questionnaires Responses Of Patients Affected By Breast Cancer</i>
---------	---

Description

This dataset contains the responses of 121 patients to 30 questions about their quality of life.

Usage

dataqol

Format

A dataframe with 121 lines and 31 columns. A line represents a patient and a column are information about the patients

Id patient Id

q1-q28 responses to 28 questions with number of levels equals to 4

q29-q30 responses to 22 questions with number of levels equals to 7

Source

The table was determined based on data associated with the package available on: <https://cran.r-project.org/package=QoLR>

dataqol.classif	<i>Questionnaires Responses Of Patients Affected By Breast Cancer</i>
-----------------	---

Description

This dataset contains the responses of 40 patients to 30 questions about their quality of life. Furthermore, a variable indicates if the patient survived from the disease.

Usage

dataqol.classif

Format

A dataframe with 40 lines and 32 columns. A line represents a patient and a column are information about the patients

Id patient Id

q1-q28 responses to 28 questions with number of levels equals to 4

q29-q30 responses to 22 questions with number of levels equals to 7

death if the patient survived (1) or not (2)

Source

The table was determined based on data associated with the package available on: <https://cran.r-project.org/package=QoLR>

Msimulated	<i>Matrix of simulated ordinal data</i>
------------	---

Description

This is a toy dataset for running simple examples.

Usage

Msimulated

Format

An ordinal data matrix with 60 lines and 50 columns. Number of levels is equal to 3. 4 blocks are simulated with (mu,pi) parameters equal to (3,0.5), (2,0.7), (1,0.8) and (2,0.6).

pejSim	<i>pejSim</i>
--------	---------------

Description

This function computes the probability for a level e_j to be sampled from a BOS distribution of parameters (μ, π) , with a number of levels equals to m . Can be used to generate data with BOS distribution.

Usage

pejSim(e_j, m, μ, π)

Arguments

e_j	levels to be sampled
m	Number of levels.
μ	mu parameter for BOS distribution.
π	pi parameter for BOS distribution.

Value

Return the probability of e_j to be sampled from a BOS distribution of parameters (μ, π) , with a number of levels equals to m .

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
library(ordinalClust)
data("dataqol")
set.seed(5)

m=7
nr=10000
mu=5
pi=0.5

probaBOS=rep(0,m)
for (im in 1:m) probaBOS[im]=pejSim(im,m,mu,pi)
M <- sample(1:m,nr,prob = probaBOS, replace=TRUE)
```

plot

*~~ Methods for Function plot in Package **ordinalClust** ~~*

Description

Plots the result of a classification, clustering or co-clustering that were performed from the following functions: bosclassif,bosclust,boscoclust.

Methods

```
signature(object = "ResultClassifOrdinal")
signature(object = "ResultClustOrdinal")
signature(object = "ResultCoclustOrdinal")
```

predict

*~~ Methods for Function predict in Package **stats** ~~*

Description

*~~ Methods for function predict in package **stats** ~~*

Methods

```
signature(object = "ResultClassifOrdinal") Use this method with the result of the function bosclassif, and a new sample to predict the classes.
```

summary

*~~ Methods for Function summary in Package **ordinalClust** ~~*

Description

Prints the result of a classification, clustering or co-clustering that were performed from the following functions: `bosclassif`, `bosclust`, `boscoclust`.

Methods

`signature(object = "ResultClassifOrdinal")`

`signature(object = "ResultClustOrdinal")`

`signature(object = "ResultCoclustOrdinal")`

Index

*Topic **datasets**

- dataqol, [9](#)
- dataqol.classif, [9](#)
- Msimulated, [10](#)

*Topic **methods**

- plot, [11](#)
- predict, [11](#)
- summary, [12](#)

- bosclassif, [2](#)
- bosclust, [4](#)
- boscoclust, [6](#)

- dataqol, [9](#)
- dataqol.classif, [9](#)

- Msimulated, [10](#)

- pejSim, [10](#)
- plot, [11](#)
- plot, ResultClassifOrdinal-method
(plot), [11](#)
- plot, ResultClustOrdinal-method (plot),
[11](#)
- plot, ResultCoclustOrdinal-method
(plot), [11](#)
- predict, [11](#)
- predict, ANY-method (predict), [11](#)
- predict, ResultClassifOrdinal-method
(predict), [11](#)
- summary, [12](#)
- summary, ResultClassifOrdinal-method
(summary), [12](#)
- summary, ResultClustOrdinal-method
(summary), [12](#)
- summary, ResultCoclustOrdinal-method
(summary), [12](#)