Package 'HDclassif'

August 23, 2023

Encoding UTF-8

Type Package

Title High Dimensional Supervised Classification and Clustering

Version 2.2.1

Depends grDevices, utils, graphics, stats, MASS

Imports rARPACK

Author Laurent Berge, Charles Bouveyron and Stephane Girard

Maintainer Laurent Berge <laurent.berge@u-bordeaux.fr>

Description Discriminant analysis and data clustering methods for high dimensional data, based on the assumption that high-dimensional data live in different subspaces with low dimensionality proposing a new parametrization of the Gaussian mixture model which combines the ideas of dimension reduction and constraints on the model.

License GPL-2

LazyLoad yes

ZipData no

RoxygenNote 7.2.3

NeedsCompilation no

Repository CRAN

Date/Publication 2023-08-23 07:50:02 UTC

R topics documented:

HDclassif-package 2
Crabs 3
demo_hddc 4
hdda 4
hddc 9
hdmda 15
plot.hdc 18
predict.hdc 19

predict.hdmda 21
setHDclassif.show 22
simuldata 23
slopeHeuristic 25
wine 26

Index **27**

`HDclassif-package` *High Dimensional Discriminant Analysis and Data Clustering*

Description

Disciminant analysis and data clustering methods for high dimensional data, based on the asumption that high-dimensional data live in different subspaces with low dimensionality, proposing a new parametrization of the Gaussian mixture model which combines the ideas of dimension reduction and constraints on the model.

Details

Package:	HDclassif
Type:	Package
Version:	2.1.0
Date:	2018-05-11
License:	GPL-2
LazyLoad:	yes

This package is used to make efficient supervised and unsupervised classification with high dimensional data. The supervised method uses the *hdda* function to get the data parameters and the *predict* function to realise the class prediction of a dataset. The unsupervised method is implemented in the *hddc* function, and once the parameters are estimated, the *predict* gives the class prediction of other datasets. The method used in the hddc is based on the Expectation - Maximisation algorithm.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

Maintainer: Laurent Berge <laurent.berge at uni.lu>

References

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High Dimensional Discriminant Analysis", *Communications in Statistics: Theory and Methods*, vol. **36** (14), pp. 2607–2623

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High-Dimensional Data Clustering", *Computational Statistics and Data Analysis*, vol. **52** (1), pp. 502–519

Berge, L. Bouveyron, C. and Girard, S. (2012) "HDclassif: An R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data", *Journal of Statistical Software*, **46**(6), 1–29, url: doi:10.18637/jss.v046.i06

`Crabs` *Morphological Measurements on Leptograpsus Crabs.*

Description

The Crabs data frame has 200 rows and 6 columns, describing 5 morphological measurements on 50 crabs each of two colour forms and both sexes, of the species Leptograspus Variegatus collected at Fremantle, W. Australia.

Usage

```
data(Crabs)
```

Format

A data frame with 200 observations on the following 6 variables.

`class` Type of the crabs: the first character represents the species - "B" or "O" for blue or orange-, the second represents the sex -"M" or "F" for male or female-.

`FL` Frontal lob size (mm).

`RW` Rear width (mm).

`CL` Carapace length (mm).

`CW` Carapace width (mm).

`BD` Body depth (mm).

Details

This dataset can also be found in the MASS package, the unique difference is the class vector which is easier to use here.

Source

Campbell, N. A. and Mahon, R. J. (1974) "A multivariate study of variation on two species of rock crab of genus Leptograspus", *Australian Journal of Zoology*, **22**, 417–425.

References

Venables, W. N. and Ripley, B. D. (2002) "Modern Applied Statistics with S". Fourth edition. Springer.

demo_hddc *Demonstration of the clustering process of HDDC.*

Description

This demonstration uses a PCA on the first two principal axis of the Crabs dataset -that can be found in the package- to show the clustering process of HDDC. At each step of the clustering, the means and directions are shown by, respectively, points and lines. This function should only be used in demo(hddc).

Usage

```
demo_hddc()
```

Value

The plots of the clustering process.

Note

The algorithm and the initialization are interactively chosen.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

See Also

hddc.

hdda *High Dimensional Discriminant Analysis*

Description

HDDA is a model-based discriminant analysis method assuming each class of the dataset live in a proper Gaussian subspace which is much smaller than the original one, the hdda.learn function calculates the parameters of each subspace in order to predict the class of new observation of this kind.

Usage

```
hdda(
  data,
  cls,
  model = "AkjBkQkDk",
  graph = FALSE,
  d_select = "Cattell",
  threshold = 0.2,
  com_dim = NULL,
  show = getHDclassif.show(),
  scaling = FALSE,
  cv.dim = 1:10,
  cv.threshold = c(0.001, 0.005, 0.05, 1:9 * 0.1),
  cv.vfold = 10,
  LOO = FALSE,
  noise.ctrl = 1e-08,
  d
)
```

Arguments

data	A matrix or a data frame of observations, assuming the rows are the observations and the columns the variables. Note that NAs are not allowed.
cls	The vector of the class of each observations, its type can be numeric or string.
model	A character string vector, or an integer vector indicating the models to be used. The available models are: "AkjBkQkDk" (default), "AkBkQkDk", "ABkQkDk", "AkjBQkDk", "AkBQkDk", "ABQkDk", "AkjBkQkD", "AkBkQkD", "ABkQkD", "AkjBQkD", "AkBQkD", "ABQkD", "AjBQD", "ABQD". It is not case sensitive and integers can be used instead of names, see details for more information. Several models can be used, if it is, only the results of the one which maximizes the BIC criterion is kept. To run all models, use model="ALL".
graph	It is for comparison sake only, when several estimations are run at the same time (either when using several models, or when using cross-validation to select the best dimension/threshold). If graph = TRUE, the plot of the results of all estimations is displayed. Default is FALSE.
d_select	Either "Cattell" (default), "BIC" or "CV". See details for more information. This parameter selects which method to use to select the intrinsic dimensions.
threshold	A float stricly within 0 and 1. It is the threshold used in the Cattell's Scree-Test.
com_dim	It is used only for common dimensions models. The user can give the common dimension s/he wants. If used, it must be an integer. Its default is set to NULL.
show	Single logical. To diplay summary information on the results after the algorithm is done: set it to TRUE. By default it takes the value of getHDclassif.show which is FALSE at the loading of the package. To permanently have show=TRUE, use setHDclassif.show(TRUE).
scaling	Logical: whether to scale the dataset (mean=0 and standard-error=1 for each variable) or not. By default the data is not scaled.

`cv.dim` A vector of integers. Only when d_select="CV". Gives the dimensions for which the CV is to be done. Note that if some dimensions are greater than what it is possible to have, those are taken off.

`cv.threshold` A vector of floats strictly within 0 and 1. Only when d_select="CV". Gives the thresholds for which the CV is to be done.

`cv.vfold` An integer. Only when d_select="CV". It gives the number of different subsamples in which the dataset is split. If "cv.vfold" is greater than the number of observations, then the program equalize them.

`LOO` If TRUE, it returns the results (classes and posterior probabilities) for leave-one-out cross-validation.

`noise.ctrl` This parameter avoids to have a too low value of the 'noise' parameter b. It garantees that the dimension selection process do not select too many dimensions (which leads to a potential too low value of the noise parameter b). When selecting the intrinsic dimensions using Cattell's scree-test or BIC, the function doesn't use the eigenvalues inferior to noise.ctrl, so that the intrinsic dimensions selected can't be higher or equal to the order of these eigenvalues.

`d` DEPRECATED. This parameter is kept for retro compatibility. Now please use the parameter d_select.

Details

Some information on the signification of the model names:

Akj are the parameters of the classes subspaces:
- if Akj: each class has its parameters and there is one parameter for each dimension
- if Ak: the classes have different parameters but there is only one per class
- if Aj: all the classes have the same parameters for each dimension (it's a particular case with a common orientation matrix)
- if A: all classes have the same one parameter

Bk are the noises of the classes subspaces:
- If Bk: each class has its proper noise
- if B: all classes have the same noise

Qk is the orientation matrix of each class:
- if Qk: all classes have its proper orientation matrix
- if Q: all classes have the same orientation matrix

Dk is the intrinsic dimension of each class:
- if Dk: the dimensions are free and proper to each class
- if D: the dimension is common to all classes

The model "all" will compute all the models, give their BIC and keep the model with the highest BIC value. Instead of writing the model names, they can also be specified using an integer. 1 represents the most general model ("AkjBkQkDk") while 14 is the most constrained ("ABQD"), the others number/name matching are given below. Note also that several models can be run at once, by using a vector of models (e.g. model = c("AKBKQKD","AKJBQKDK","AJBQD") is equivalent to model = c(8,4,13); to run the 6 first models, use model=1:6). If all the models are to be run, model="all" is faster than model=1:14.

AkjBkQkDk	1	AkjBkQkD	7

AkBkQkDk	2	AkBkQkD	8
ABkQkDk	3	ABkQkD	9
AkjBQkDk	4	AkjBQkD	10
AkBQkDk	5	AkBQkD	11
ABQkDk	6	ABQkD	12
AjBQD	13	ABQD	14

The parameter d, is used to select the intrinsic dimensions of the subclasses. Here are his definictions:

- "Cattell": The Cattell's scree-test is used to gather the intrinsic dimension of each class. If the model is of common dimension (models 7 to 14), the scree-test is done on the covariance matrix of the whole dataset.
- "BIC": The intrinsic dimensions are selected with the BIC criterion. See Bouveyron *et al.* (2010) for a discussion of this topic. For common dimension models, the procedure is done on the covariance matrix of the whole dataset.
- "CV": A V-fold cross-validation (CV) can be done in order to select the best threshold (for all models) or the best common dimensions (models 7 to 14). The V-fold cross-validation is done for each dimension (respectively threshold) in the argument "cv.dim" (resp. "cv.threshold"), then the dimension (resp. threshold) that gives the best good classification rate is kept. The dataset is split in "cv.vfold" (default is 10) *random* subsamples, then CV is done for each sample: each of them is used as validation data while the remaining data is used as training data. For sure, if "cv.vfold" equals the number of observations, then this CV is equivalent to a leave-one-out.

Value

hdda returns an 'hdc' object; it's a list containing:

`model`	The name of the model.
`k`	The number of classes.
`d`	The dimensions of each class.
`a`	The parameters of each class subspace.
`b`	The noise of each class subspace.
`mu`	The mean of each variable for each class.
`prop`	The proportion of each class.
`ev`	The eigen values of the var/covar matrix.
`Q`	The orthogonal matrix of orientation of each class.
`kname`	The name of each class.
`BIC`	The BIC value of the model used.
`scaling`	The centers and the standard deviation of the original dataset.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

References

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High Dimensional Discriminant Analysis", *Communications in Statistics: Theory and Methods*, vol. **36** (14), pp. 2607–2623

Bouveyron, C. Celeux, G. and Girard, S. (2010) "Intrinsic dimension estimation by maximum likelihood in probabilistic PCA", Technical Report 440372, Universite Paris 1 Pantheon-Sorbonne

Berge, L. Bouveyron, C. and Girard, S. (2012) "HDclassif: An R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data", *Journal of Statistical Software*, **46**(6), 1–29, url: doi:10.18637/jss.v046.i06

See Also

hddc, predict.hdc, plot.hdc

Examples

```
# Example 1:
data<-simuldata(1000, 1000, 50, K=5)
X <- data$X
clx <- data$clx
Y <- data$Y
cly <- data$cly
# we get the HDDA parameters:
prms1 <- hdda(X, clx)

cl1 <- predict(prms1, Y, cly)
# the class vector of Y estimated with HDDA:
cl1$class

# another model is used:
prms1 <- hdda(X, clx, model=12)
#model=12 is equivalent to model="ABQkD"
cl1 <- predict(prms1, Y, cly)

# Example 2:
data(wine)
a <- wine[,-1]
z <- wine[,1]
prms2 <- hdda(a, z, model='all', scaling=TRUE, d_select="bic", graph=TRUE)
cl2 <- predict(prms2, a, z)

# getting the best dimension
# using a common dimension model
# we do LOO-CV using cv.vfold=nrow(a)
prms3 <- hdda(a, z, model="akjbkqkd", d_select="CV", cv.vfold=nrow(a), scaling=TRUE, graph=TRUE)

cl3 <- predict(prms3, a, z)

# Example 3:
# Validation with LOO
prms4 = hdda(a, z, LOO=TRUE, scaling=TRUE)
sum(prms4$class==z) / length(z)
```

hddc *High Dimensional Data Clustering*

Description

HDDC is a model-based clustering method. It is based on the Gaussian Mixture Model and on the idea that the data lives in subspaces with a lower dimension than the dimension of the original space. It uses the Expectation - Maximisation algorithm to estimate the parameters of the model.

Usage

```
hddc(
  data,
  K = 1:10,
  model = c("AkjBkQkDk"),
  threshold = 0.2,
  criterion = "bic",
  com_dim = NULL,
  itermax = 200,
  eps = 0.001,
  algo = "EM",
  d_select = "Cattell",
  init = "kmeans",
  init.vector,
  show = getHDclassif.show(),
  mini.nb = c(5, 10),
  scaling = FALSE,
  min.individuals = 2,
  noise.ctrl = 1e-08,
  mc.cores = 1,
  nb.rep = 1,
  keepAllRes = TRUE,
  kmeans.control = list(),
  d_max = 100,
  subset = Inf,
  d
)
```

Arguments

`data`	A matrix or a data frame of observations, assuming the rows are the observations and the columns the variables. Note that NAs are not allowed.
`K`	A vector of integers specifying the number of clusters for which the BIC and the parameters are to be calculated; the function keeps the parameters which maximises the `criterion`. Default is 1:10.

`model` A character string vector, or an integer vector indicating the models to be used. The available models are: "AkjBkQkDk" (default), "AkBkQkDk", "ABkQkDk", "AkjBQkDk", "AkBQkDk", "ABQkDk", "AkjBkQkD", "AkBkQkD", "ABkQkD", "AkjBQkD", "AkBQkD", "ABQkD", "AjBQD", "ABQD". It is not case sensitive and integers can be used instead of names, see details for more information. Several models can be used, if it is, only the results of the one which maximizes the BIC criterion is kept. To run all models, use model="ALL".

`threshold` A float stricly within 0 and 1. It is the threshold used in the Cattell's Scree-Test.

`criterion` Either "BIC" or "ICL". If several models are run, the best model is selected using the criterion defined by `criterion`.

`com_dim` It is used only for common dimensions models. The user can give the common dimension s/he wants. If used, it must be an integer. Its default is set to NULL.

`itermax` The maximum number of iterations allowed. The default is 200.

`eps` A positive double, default is 0.001. It is the stopping criterion: the algorithm stops when the difference between two successive log-likelihoods is lower than `eps`.

`algo` A character string indicating the algorithm to be used. The available algorithms are the Expectation-Maximisation ("EM"), the Classification E-M ("CEM") and the Stochastic E-M ("SEM"). The default algorithm is the "EM".

`d_select` Either "Cattell" (default) or "BIC". See details for more information. This parameter selects which method to use to select the intrinsic dimensions.

`init` A character string or a vector of clusters. It is the way to initialize the E-M algorithm. There are five possible initialization: "kmeans" (default), "param", "random", "mini-em" or "vector". See details for more information. It can also be directly initialized with a vector containing the prior classes of the observations. If `init = "vector"`, then you should add the argument `init.vector`.

`init.vector` A vector of integers or factors. It is a user-given initialization. It should be of the same length as of the data. Only used when `init = "vector"`.

`show` Single logical. To diplay summary information on the results after the algorithm is done: set it to `TRUE`. By default it takes the value of `getHDclassif.show` which is FALSE at the loading of the package. To permanently have `show=TRUE`, use `setHDclassif.show(TRUE)`.

`mini.nb` A vector of integers of length two. This parameter is used in the "mini-em" initialization. The first integer sets how many times the algorithm is repeated; the second sets the maximum number of iterations the algorithm will do each time. For example, if `init="mini-em"` and `mini.nb=c(5,10)`, the algorithm wil be lauched 5 times, doing each time 10 iterations; finally the algorithm will begin with the initialization that maximizes the log-likelihood.

`scaling` Logical: whether to scale the dataset (mean=0 and standard-error=1 for each variable) or not. By default the data is not scaled.

`min.individuals`
Positive integer greater than 2 (default). This parameter is used to control for the minimum population of a class. If the population of a class becomes stricly inferior to 'min.individuals' then the algorithm stops and gives the message: 'pop<min.indiv.'. Here the meaning of "population of a class" is the sum of its posterior probabilities. The value of 'min.individuals' cannot be lower than 2.

`noise.ctrl` This parameter avoids to have a too low value of the 'noise' parameter b. It garantees that the dimension selection process do not select too many dimensions (which leads to a potential too low value of the noise parameter b). When selecting the intrinsic dimensions using Cattell's scree-test or BIC, the function doesn't use the eigenvalues inferior to noise.ctrl, so that the intrinsic dimensions selected can't be higher or equal to the order of these eigenvalues.

`mc.cores` Positive integer, default is 1. If `mc.cores>1`, then parallel computing is used, using `mc.cores` cores. Warning for Windows users only: the parallel computing can sometimes be slower than using one single core (due to how `parLapply` works).

`nb.rep` A positive integer (default is 1). Each estimation (i.e. combination of (model, K, threshold)) is repeated `nb.rep` times and only the estimation with the highest log-likelihood is kept.

`keepAllRes` Logical. Should the results of all runs be kept? If so, an argument `all_results` is created in the results. Default is `TRUE`.

`kmeans.control` A list. The elements of this list should match the parameters of the kmeans initialization (see `kmeans` help for details). The parameters are "iter.max", "nstart" and "algorithm".

`d_max` A positive integer. The maximum number of dimensions to be computed. Default is 100. It means that the instrinsic dimension of any cluster cannot be larger than `d_max`. It quickens a lot the algorithm for datasets with a large number of variables (e.g. thousands).

`subset` An positive integer, default is `Inf`. In case of large data sets it might be useful to perform HDDC on a subsample of the data: this is the use of this argument. If `subset` is to a value smaller than the number of observations of the dataset then: HDDC is performed on a random subsample of size `subset` and once a clustering is obtained on this subsample, the posterior of the clustering is computed on the full sample.

`d` DEPRECATED. This parameter is kept for retro compatibility. Now please use the parameter d_select.

Details

Some information on the signification of the model names:

Akj are the parameters of the classes subspaces:
- if Akj: each class has its parameters and there is one parameter for each dimension
- if Ak: the classes have different parameters but there is only one per class
- if Aj: all the classes have the same parameters for each dimension (it's a particular case with a common orientation matrix)
- if A: all classes have the same one parameter

Bk are the noises of the classes subspaces:
- If Bk: each class has its proper noise
- if B: all classes have the same noise

Qk is the orientation matrix of each class:
- if Qk: all classes have its proper orientation matrix
- if Q: all classes have the same orientation matrix

Dk is the intrinsic dimension of each class:
- if Dk: the dimensions are free and proper to each class
- if D: the dimension is common to all classes

The model "ALL" will compute all the models, give their BIC and keep the model with the highest BIC value. Instead of writing the model names, they can also be specified using an integer. 1 represents the most general model ("AkjBkQkDk") while 14 is the most constrained ("ABQD"), the others number/name matching are given below. Note also that several models can be run at once, by using a vector of models (e.g. model = c("AKBKQKD","AKJBQKDK","AJBQD") is equivalent to model = c(8,4,13); to run the 6 first models, use model=1:6). If all the models are to be run, model="all" is faster than model=1:14.

AkjBkQkDk	1	AkjBkQkD	7
AkBkQkDk	2	AkBkQkD	8
ABkQkDk	3	ABkQkD	9
AkjBQkDk	4	AkjBQkD	10
AkBQkDk	5	AkBQkD	11
ABQkDk	6	ABQkD	12
AjBQD	13	ABQD	14

The parameter `d_select`, is used to select the intrinsic dimensions of the subclasses. Here are its definitions:

- "Cattell": The Cattell's scree-test is used to gather the intrinsic dimension of each class. If the model is of common dimension (models 7 to 14), the scree-test is done on the covariance matrix of the whole dataset.
- "BIC": The intrinsic dimensions are selected with the BIC criterion. See Bouveyron *et al.* (2010) for a discussion of this topic. For common dimension models, the procedure is done on the covariance matrix of the whole dataset.
- Note that "Cattell" (resp. "BIC") can be abreviated to "C" (resp. "B") and that this argument is not case sensitive.

The different initializations are:

"param": it is initialized with the parameters, the means being generated by a multivariate normal distribution and the covariance matrix being common to the whole sample

"mini-em": it is an initialization strategy, the classes are randomly initialized and the E-M algorithm makes several iterations, this action is repetead a few times (the default is 5 iterations and 10 times), at the end, the initialization choosen is the one which maximise the log-likelihood (see mini.nb for more information about its parametrization)

"random": the classes are randomly given using a multinomial distribution

"kmeans": the classes are initialized using the kmeans function (with: algorithm="Hartigan-Wong"; nstart=4; iter.max=50); note that the user can use his own arguments for kmeans using the dot-dot-dot argument

A prior class vector: It can also be directly initialized with a vector containing the prior classes of the observations. To do so use `init="vector"` and provide the vector in the argument `init.vector`.

The BIC criterion used in this function is to be maximized and is defined as 2*LL-k*log(n) where LL is the log-likelihood, k is the number of parameters and n is the number of observations.

Value

hddc returns an 'hdc' object; it's a list containing:

`model`	The name of the model.
`K`	The number of classes.
`d`	The dimensions of each class.
`a`	The parameters of each class subspace.
`b`	The noise of each class subspace.
`mu`	The mean of each variable for each class.
`prop`	The proportion of each class.
`ev`	The eigen values of the var/covar matrix.
`Q`	The orthogonal matrix of orientation of each class.
`loglik`	The log-likelihood.
`loglik_all`	The log-likelihood of all iterations. Note that if `subset` was used, then this vector represents the likelihoods evaluations for the subsample on which HDDC was performed (i.e. not the likelihood for the full dataset – so these values are smaller than the on given in 'loglik' which concerns the whole sample after the estimation).
`posterior`	The matrix of the probabilities to belong to a class for each observation and each class.
`class`	The class vector obtained by the clustering.
`com_ev`	Only if this is a common dimension model. The eigenvalues of the var/covar matrix of the whole dataset.
`N`	The number of observations.
`complexity`	The number of parameters of the model.
`threshold`	The threshold used for the Cattell scree-test.
`d_select`	The way the dimensions were selected.
`BIC`	The BIC of the model.
`ICL`	The ICL of the model.
`criterion`	The criterion used to select the model.
`call`	The call.
`allCriteria`	The data.frame with the combination (model, K, threshold) and the associated values of the likelihood (LL), BIC and ICL, as well as the rank of each of the models with respect to the selection criterion. It also reports the original order in which were estimated the models as well as each model complexity
`all_results`	Only if `keepAllRes=TRUE`. The parameters of all estimations that were run.
`scaling`	Only if `scaling=TRUE`. The centers and the standard deviation of the original dataset.
`id_subset`	Only if `subset` is used. The observation IDs of the subsample on which the HDDC parameters were estimated.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

References

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High-Dimensional Data Clustering", *Computational Statistics and Data Analysis*, vol. **52** (1), pp. 502–519

Berge, L. Bouveyron, C. and Girard, S. (2012) "HDclassif: An R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data", *Journal of Statistical Software*, **46**(6), 1–29, url: doi:10.18637/jss.v046.i06

See Also

hdda, predict.hdc, plot.hdc.

Examples

```
# Example 1:
data <- simuldata(1000, 1000, 50)
X <- data$X
clx <- data$clx
Y <- data$Y
cly <- data$cly

#clustering of the simulated dataset:
prms1 <- hddc(X, K=3, algo="CEM", init='param')

#class vector obtained by the clustering:
prms1$class

#We can look at the adjusted rand index to assess the goodness of fit
res1 <- predict(prms1, X, clx)
res2 <- predict(prms1, Y)
#the class predicted using hddc parameters on the test dataset:
res2$class


# Example 2:
data(Crabs)

# clustering of the Crabs dataset:
prms3 <- hddc(Crabs[,-1], K=4, algo="EM", init='mini-em')
res3 <- predict(prms3, Crabs[,-1], Crabs[,1])

# another example using the Crabs dataset
prms4 <- hddc(Crabs[,-1], K=1:8, model=c(1,2,7,9))

# model=c(1,2,7,9) is equivalent to:
# model=c("AKJBKQKDK","AKBKQKDK","AKJBKQKD"#' ,"ABKQKD")
res4 <- predict(prms4, Crabs[,-1], Crabs[,1])
```

```
# PARALLEL COMPUTING
## Not run:
# Same example but with Parallel Computing => platform specific
# (slower for Windows users)
# To enable it, just use the argument 'mc.cores'
prms5 <- hddc(Crabs[,-1], K=1:8, model=c(1,2,7,9), mc.cores=2)

## End(Not run)

# LARGE DATASETS
# Assume you have a very large data set
# => you can use the argument 'subset' to obtain quick results:
## Not run:
# we take a subset of 10000 observations and run hddc
# once the classification is done, the posterior is computed
# on the full data
prms = hddc(bigData, subset = 10000)
# You obtain a much faster (although less precise)
# classification of the full dataset:
table(prms$class)

## End(Not run)
```

---

hdmda *Mixture Discriminant Analysis with HD Gaussians*

Description

HD-MDA implements mixture discriminant analysis (MDA, Hastie & Tibshirani, 1996) with HD Gaussians instead of full Gaussians. Each class is assumed to be made of several class-specific groups in which the data live in low-dimensional subspaces. From a technical point of view, a clustering is done using hddc in each class.

Usage

```
hdmda(X,cls,K=1:10,model='AkjBkQkDk',show=FALSE,...)
```

Arguments

X	A matrix or a data frame of observations, assuming the rows are the observations and the columns the variables. Note that NAs are not allowed.
cls	The vector of the class of each observations, its type can be numeric or string.
K	A vector of integers specifying the number of clusters for which the BIC and the parameters are to be calculated; the function keeps the parameters which maximises the BIC. Note that the length of the vector K can't be larger than 20. Default is 1:10.

`model` A character string vector, or an integer vector indicating the models to be used. The available models are: "AkjBkQkDk" (default), "AkBkQkDk", "ABkQkDk", "AkjBQkDk", "AkBQkDk", "ABQkDk", "AkjBkQkD", "AkBkQkD", "ABkQkD", "AkjBQkD", "AkBQkD", "ABQkD", "AjBQD", "ABQD". It is not case sensitive and integers can be used instead of names, see details for more information. Several models can be used, if it is, only the results of the one which maximizes the BIC criterion is kept. To run all models, use model="ALL".

`show` Use show = TRUE to display some information related to the clustering.

`...` Any argument that can be used by the function `hddc`.

Details

Some information on the signification of the model names:

Akj are the parameters of the classes subspaces:
- if Akj: each class has its parameters and there is one parameter for each dimension
- if Ak: the classes have different parameters but there is only one per class
- if Aj: all the classes have the same parameters for each dimension (it's a particular case with a common orientation matrix)
- if A: all classes have the same one parameter

Bk are the noises of the classes subspaces:
- If Bk: each class has its proper noise
- if B: all classes have the same noise

Qk is the orientation matrix of each class:
- if Qk: all classes have its proper orientation matrix
- if Q: all classes have the same orientation matrix

Dk is the intrinsic dimension of each class:
- if Dk: the dimensions are free and proper to each class
- if D: the dimension is common to all classes

The model "all" will compute all the models, give their BIC and keep the model with the highest BIC value. Instead of writing the model names, they can also be specified using an integer. 1 represents the most general model ("AkjBkQkDk") while 14 is the most constrained ("ABQD"), the others number/name matching are given below:

AkjBkQkDk	1	AkjBkQkD	7
AkBkQkDk	2	AkBkQkD	8
ABkQkDk	3	ABkQkD	9
AkjBQkDk	4	AkjBQkD	10
AkBQkDk	5	AkBQkD	11
ABQkDk	6	ABQkD	12
AjBQD	13	ABQD	14

Value

`hdmda` returns an 'hdmda' object which is a list containing:

`alpha` Estimated prior probabilities for the classes.

prms	Estimated mixture parameters for each class.
kname	The name (level) of each class.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

References

C. Bouveyron and C. Brunet (2014), "Model-based clustering of high-dimensional data: A review", Computational Statistics and Data Analysis, vol. 71, pp. 52-78.

Bouveyron, C. Girard, S. and Schmid, C. (2007), "High Dimensional Discriminant Analysis", Communications in Statistics: Theory and Methods, vol. 36 (14), pp. 2607-2623.

Bouveyron, C. Celeux, G. and Girard, S. (2011), "Intrinsic dimension estimation by maximum likelihood in probabilistic PCA", Pattern Recognition Letters, vol. 32 (14), pp. 1706-1713.

Berge, L. Bouveyron, C. and Girard, S. (2012), "HDclassif: An R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data", Journal of Statistical Software, 46(6), pp. 1-29, url: doi:10.18637/jss.v046.i06.

Hastie, T., & Tibshirani, R. (1996), "Discriminant analysis by Gaussian mixtures", Journal of the Royal Statistical Society, Series B (Methodological), pp. 155-176.

See Also

hdda, hddc

Examples

```
# Load the Wine data set
data(wine)
cls = wine[,1]; X = scale(wine[,-1])

# A simple use...
out = hdmda(X[1:100,],cls[1:100])
res = predict(out,X[101:nrow(X),])

# Comparison between hdmda and hdda in a CV setup
set.seed(123); nb = 10; Err = matrix(NA,2,nb)
for (i in 1:nb){
  cat('.')
  test = sample(nrow(X),50)
  out0 = lda(X[-test,],cls[-test])
  res0 = predict(out0,X[test,])
  Err[1,i] = sum(res0$class != cls[test]) / length(test)
  out = hdmda(X[-test,],cls[-test],K=1:3,model="AKJBQKDK")
  res = predict(out,X[test,])
  Err[2,i] = sum(res$class != cls[test]) / length(test)
}
cat('\n')
boxplot(t(Err),names=c('LDA','HD-MDA'),col=2:3,ylab="CV classifciation error",
  main='CV classifciation error on Wine data')
```

plot.hdc *Cattell's Scree-Test for 'hdc' class objects.*

Description

This function plots Cattell's scree-test or the BIC selection, using parameters coming from *hdda* or *hddc* functions.

Usage

```
## S3 method for class 'hdc'
plot(x, method = NULL, threshold = NULL, noise.ctrl=1e-8, ...)
```

Arguments

x	A 'hdc' class object obtained using *hdda* or *hddc* methods.
method	The method used to select the intrinsic dimension. It can be "BIC" or "Cattell". By default it takes the method used when obtaining the parameters using *hdda* or *hddc*. Note that "Cattell" (resp. "BIC") can be abreviated to "C" (resp. "B") and that this argument is not case sensitive.
threshold	The threshold used in Cattell's Scree-Test. By default it takes the threshold in the argument x, if none, the default value of the threshold is 0.2.
noise.ctrl	This parameter avoids to have a too low value of the 'noise' parameter b. It garantees that the dimension selection process do not select too many dimensions (which leads to a potential too low value of the noise parameter b). When selecting the intrinsic dimensions using Cattell's scree-test or BIC, the function doesn't use the eigenvalues inferior to noise.ctrl, so that the intrinsic dimensions selected can't be higher or equal to the order of these eigenvalues.
...	Arguments based from or to other methods.

Value

If method = "Cattell" The plot of the eigen values and of the sequential differences of the eigen values. The dimension to retain is the one before the last fall of the eigenvalues' differences below the threshold.

If method = "BIC" The BIC related to the dimension for each class. It stops after the first fall of the BIC.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

References

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High Dimensional Discriminant Analysis", *Communications in Statistics: Theory and Methods*, vol. **36** (14), pp. 2607–2623

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High-Dimensional Data Clustering", *Computational Statistics and Data Analysis*, vol. **52** (1), pp. 502–519

Berge, L. Bouveyron, C. and Girard, S. (2012) "HDclassif: An R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data", *Journal of Statistical Software*, **46**(6), 1–29, url: doi:10.18637/jss.v046.i06

See Also

hdda, hddc, predict.hdc.

Examples

```
# Example 1 :
data(wine)
a <- wine[,-1]
z <- wine[,1]

prms1 <- hdda(a, z, model="AkBkQkDk", scaling=TRUE, d_select="bic")

#the plot related to the selection that has been done: BIC
plot(prms1)

#it shows the plot of Cattell's scree-test, with a threshold of .3
plot(prms1,"Cattell",0.3)


prms2 <- hdda(a, z, model="AkBkQkD", scaling=TRUE, d_select="cattell")
#the plot related to the selection that has been done: Cattell's scree-test
plot(prms2)
#the plot of the BIC
plot(prms2,"b")
```

---

predict.hdc *Prediction method for 'hdc' class objects.*

Description

This function computes the class prediction of a dataset with respect to the model-based supervised and unsupervised classification methods hdda and hddc.

Usage

```
## S3 method for class 'hdc'
predict(object, data, cls = NULL, ...)
```

Arguments

`object`	An 'hdc' class object obtained by using hdda or hddc function.
`data`	A matrix or a data frame of observations, assuming the rows are the observations and the columns the variables. The data should be in the exact same format as the one that trained the model. Note that NAs are not allowed.
`cls`	A vector of the thue classes of each observation. It is optional and used to be compared to the predicted classes, default is NULL.
`...`	Not currently used.

Value

`class`	vector of the predicted class.
`prob`	The matrix of the probabilities to belong to a class for each observation and each class.
`loglik`	The likelihood of the classification on the new data.

If the initial class vector is given to the argument 'cls' then the adjusted rand index (ARI) is also returned. Also the following object is returned:

`ARI`	The confusion matrix of the classification.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

References

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High Dimensional Discriminant Analysis", *Communications in Statistics: Theory and Methods*, vol. **36** (14), pp. 2607–2623

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High-Dimensional Data Clustering", *Computational Statistics and Data Analysis*, vol. **52** (1), pp. 502–519

Berge, L. Bouveyron, C. and Girard, S. (2012) "HDclassif: An R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data", *Journal of Statistical Software*, **46**(6), 1–29, url: doi:10.18637/jss.v046.i06

See Also

The functions to do high dimensional classification hdda or clustering hddc.

Examples

```
# Example 1:
data <- simuldata(1000, 1000, 50)
X <- data$X
clx <- data$clx
Y <- data$Y
cly <- data$cly
```

```
#clustering of the gaussian dataset:
prms1 <- hddc(X, K=3, algo="CEM", init='param')

#class vector obtained by the clustering:
prms1$class

# only to see the good classification rate and
# the Adjusted Rand Index:
res1 <- predict(prms1, X, clx)
res2 <- predict(prms1, Y)

#the class predicted using hddc parameters on the test dataset:
res2$class


# Example 2:
data(Crabs)
#clustering of the Crabs dataset:
prms3 <- hddc(Crabs[,-1], K=4, algo="EM", init='kmeans')
res3 <- predict(prms3, Crabs[,-1], Crabs[,1])
```

`predict.hdmda` *Prediction method for 'hdmda' class objects.*

Description

This function computes the class prediction of a dataset with respect to the model-based supervised classification method `hdmda`.

Usage

```
## S3 method for class 'hdmda'
predict(object, X, ...)
```

Arguments

`object`	An object of class 'hdmda'.
`X`	A matrix or a data frame of observations, assuming the rows are the observations and the columns the variables. Note that NAs are not allowed.
`...`	Arguments based from or to other methods. Not currently used.

Value

`class`	vector of the predicted class.
`posterior`	The matrix of the probabilities to belong to a class for each observation and each class.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

References

C. Bouveyron and C. Brunet (2014), "Model-based clustering of high-dimensional data: A review", Computational Statistics and Data Analysis, vol. 71, pp. 52-78.

Bouveyron, C. Girard, S. and Schmid, C. (2007), "High Dimensional Discriminant Analysis", Communications in Statistics: Theory and Methods, vol. 36 (14), pp. 2607-2623.

Bouveyron, C. Celeux, G. and Girard, S. (2011), "Intrinsic dimension estimation by maximum likelihood in probabilistic PCA", Pattern Recognition Letters, vol. 32 (14), pp. 1706-1713.

Berge, L. Bouveyron, C. and Girard, S. (2012), "HDclassif: An R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data", Journal of Statistical Software, 46(6), pp. 1-29, url: doi:10.18637/jss.v046.i06.

Hastie, T., & Tibshirani, R. (1996), "Discriminant analysis by Gaussian mixtures", Journal of the Royal Statistical Society, Series B (Methodological), pp. 155-176.

See Also

hdmda

Examples

```
# Load the Wine data set
data(wine)
cls = wine[,1]; X = scale(wine[,-1])

# A simple use...
out = hdmda(X[1:100,],cls[1:100])
res = predict(out,X[101:nrow(X),])
```

`setHDclassif.show` *Sets/gets the default 'show' argument in HDDC and HDDA*

Description

Sets/gets the default value for 'show' argument in HDDC and HDDC. When `TRUE` then clustering information is returned at the end of the process.

Usage

```
setHDclassif.show(show)

getHDclassif.show
```

Arguments

show	Single logical with default. Will specify the default value of the show argument in HDDA and HDDC.

Format

An object of class function of length 1.

Value

getHDclassif.show returns the default value.

Examples

```
data(Crabs)

# clustering of the Crabs dataset:
prms <- hddc(Crabs[,-1], K=4)
# By default no information is displayed

# To show information:
prms <- hddc(Crabs[,-1], K=4, show = TRUE)

# To set it permanently:
setHDclassif.show(TRUE)
prms <- hddc(Crabs[,-1], K=4)

# to disable it permanently:
setHDclassif.show(FALSE)
```

---

simuldata *Gaussian Data Generation*

Description

This function generates two datasets according to the model [AkBkQkDk] of the HDDA gaussian mixture model paramatrisation (see ref.).

Usage

```
simuldata(nlearn, ntest, p, K = 3, prop = NULL, d = NULL, a = NULL, b = NULL)
```

Arguments

nlearn	The size of the learning dataset to be generated.
ntest	The size of the testing dataset to be generated.
p	The number of variables.
K	The number of classes.
prop	The proportion of each class.
d	The dimension of the intrinsic subspace of each class.
a	The value of the main parameter of each class.
b	The noise of each class.

Value

X	The learning dataset.
clx	The class vector of the learning dataset.
Y	The test dataset.
cly	The class vector of the test dataset.
prms	The principal parameters used to generate the datasets.

Author(s)

Laurent Berge, Charles Bouveyron and Stephane Girard

References

Bouveyron, C. Girard, S. and Schmid, C. (2007) "High Dimensional Discriminant Analysis", *Communications in Statistics : Theory and Methods*, vol. **36**(14), pp. 2607–2623

See Also

hddc, hdda.

Examples

```
data <- simuldata(500, 1000, 50, K=5, prop=c(0.2,0.25,0.25,0.15,0.15))
X <- data$X
clx <- data$clx
f <- hdda(X, clx)
Y <- data$Y
cly <- data$cly
e <- predict(f, Y, cly)
```

`slopeHeuristic` *Slope Heuristic for HDDC objects*

Description

This function computes the slope heuristic for a set of objects obtained by the function hddc. The slope heuristic is a criterion in which the likelihood is penalized according to the result of the fit of the likelihoods on the complexities of the models.

Usage

```
slopeHeuristic(x, plot = FALSE)
```

Arguments

x	An hdc object, obtained from the function hddc.
plot	Logical, default is FALSE. If TRUE, then a graph representing: 1) the likelihoods, the complexity, the fit (i.e. the slope) and 2) the value of the slope heuristic (in blue squares).

Details

This function is only useful if there are many models (at least 3, better if more) that were estimated by the function hddc. If there are less than 2 models, the function wil l return an error.

Value

A list of two elements:

best_model_index	The index of the best model, among all estimated models.
allCriteria	The data.frame containing all the criteria, with the new slope heuristic.

Examples

```
# Clustering of the Crabs data set
data(Crabs)
prms = hddc(Crabs[,-1], K = 1:10) # we estimate ten models
slope = slopeHeuristic(prms, plot = TRUE)
plot(slope$allCriteria) # The best model is indeed for 4 clusters
prms$all_results[[slope$best_model_index]] # we extract the best model
```

wine *Wine dataset*

Description

These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines.

Usage

```
data(wine)
```

Format

A data frame with 178 observations on the following 14 variables :

`class` The class vector, the three different cultivars of wine are reprensented by the three integers : 1 to 3.

`V1` Alcohol

`V2` Malic acid

`V3` Ash

`V4` Alcalinity of ash

`V5` Magnesium

`V6` Total phenols

`V7` Flavanoids

`V8` Nonflavanoid phenols

`V9` Proanthocyanins

`V10` Color intensity

`V11` Hue

`V12` OD280/OD315 of diluted wines

`V13` Proline

Source

This dataset is from the UCI machine learning repository, provided here : http://archive.ics.uci.edu/ml/datasets/Wine.

Examples

```
data(wine)
```

Index

∗ **cattell**
    plot.hdc, 18
∗ **clustering**
    hddc, 9
    plot.hdc, 18
    predict.hdc, 19
∗ **datasets**
    Crabs, 3
    setHDclassif.show, 22
    wine, 26
∗ **demo**
    demo_hddc, 4
∗ **gaussian**
    simuldata, 23
∗ **generation**
    simuldata, 23
∗ **hdda**
    plot.hdc, 18
    predict.hdc, 19
∗ **hddc**
    plot.hdc, 18
    predict.hdc, 19
∗ **high-dimensional data**
    hdmda, 15
∗ **mixture discriminant analysis**
    hdmda, 15
∗ **package**
    HDclassif-package, 2

Crabs, 3

demo_hddc, 4

getHDclassif.show, *5*, *10*
getHDclassif.show *(*setHDclassif.show*)*, 22

HDclassif *(*HDclassif-package*)*, 2
HDclassif-package, 2
hdda, 4, *14*, *17*, *19*, *20*, *24*
hddc, *4*, *8*, 9, *15–17*, *19*, *20*, *24*, *25*
hdmda, 15, *21*, *22*

kmeans, *11*

parLapply, *11*
plot.hdc, *8*, *14*, 18
predict.hdc, *8*, *14*, *19*, 19
predict.hdmda, 21

setHDclassif.show, 22
simuldata, 23
slopeHeuristic, 25

wine, 26