Package 'rebmix'

December 11, 2019

Version 2.11.0

Title Finite Mixture Modeling, Clustering & Classification

Suggests flexmix, mclust, mixtools

Description R functions for random univariate and multivariate finite mixture model generation, estimation, clustering, latent class analysis and classification. Variables can be continuous, discrete, independent or dependent and may follow normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or circular von Mises parametric families.

Imports methods, stats, utils, graphics, grDevices, mvtnorm

License GPL (>= 2)

Author Marko Nagode [aut, cre],
Branislav Panic [ctb]

Maintainer Marko Nagode <marko.nagode@fs.uni-lj.si>

NeedsCompilation yes

Depends R (>= 2.10)

Repository CRAN

Date/Publication 2019-12-11 14:20:09 UTC

R topics documented:

`adult` *Adult Dataset*

Description

The `adult` dataset containing 48842 instances with 16 continuous, binary and discrete variables was extracted from the census bureau database. Extraction was done by Barry Becker from the 1994 census bureau database.

Usage

```
data("adult")
```

Format

adult is a data frame with 48842 cases (rows) and 16 variables (columns) named:

1. Type binary train or test.
2. Age continuous.
3. Workclass one of the 8 discrete values private, self-emp-not-inc, self-emp-inc, federal-gov, local-gov, state-gov, without-pay or never-worked.
4. Fnlwgt stands for continuous final weight.
5. Education one of the 16 discrete values bachelors, some-college, 11th, hs-grad, prof-school, assoc-acdm, assoc-voc, 9th, 7th-8th, 12th, masters, 1st-4th, 10th, doctorate, 5th-6th or preschool.
6. Education.Num continuous.
7. Marital.Status one of the 7 discrete values married-civ-spouse, divorced, never-married, separated, widowed, married-spouse-absent or married-af-spouse.
8. Occupation one of the 14 discrete values tech-support, craft-repair, other-service, sales, exec-managerial, prof-specialty, handlers-cleaners, machine-op-inspct, adm-clerical, farming-fishing, transport-moving, priv-house-serv, protective-serv or armed-forces.
9. Relationship one of the 6 discrete values wife, own-child, husband, not-in-family, other-relative or unmarried.
10. Race one of the 5 discrete values white, asian-pac-islander, amer-indian-eskimo, other or black.
11. Sex binary female or male.
12. Capital.Gain continuous.
13. Capital.Loss continuous.
14. Hours.Per.Week continuous.
15. Native.Country one of the 41 discrete values united-states, cambodia, england, puerto-rico, canada, germany, outlying-us(guam-usvi-etc), india, japan, greece, south, china, cuba, iran, honduras, philippines, italy, poland, jamaica, vietnam, mexico, portugal, ireland, france, dominican-republic, laos, ecuador, taiwan, haiti, columbia, hungary, guatemala, nicaragua, scotland, thailand, yugoslavia, el-salvador, trinadad&tobago, peru, hong or holand-netherlands.
16. Income binary <=50k or >50k.

Source

A. Asuncion and D. J. Newman. Uci machine learning repository, 2007. http://archive.ics.uci.edu/ml.

References

A. Asuncion and D. J. Newman. Uci machine learning repository, 2007. http://archive.ics.uci.edu/ml.

Examples

```
data("adult")

# Find complete cases.

adult <- adult[complete.cases(adult),]

# Show level attributes for binary and discrete variables.

levels(adult[["Type"]])
levels(adult[["Workclass"]])
levels(adult[["Education"]])
levels(adult[["Marital.Status"]])
levels(adult[["Occupation"]])
levels(adult[["Relationship"]])
levels(adult[["Race"]])
levels(adult[["Sex"]])
levels(adult[["Native.Country"]])
levels(adult[["Income"]])
```

AIC-methods *Akaike Information Criterion*

Description

Returns the Akaike information criterion at pos.

Usage

```
## S4 method for signature 'REBMIX'
AIC(x = NULL, pos = 1, ...)
## S4 method for signature 'REBMIX'
AIC3(x = NULL, pos = 1, ...)
## S4 method for signature 'REBMIX'
AIC4(x = NULL, pos = 1, ...)
## S4 method for signature 'REBMIX'
AICc(x = NULL, pos = 1, ...)
## S4 method for signature 'REBMIX'
CAIC(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

`signature(x = "REBMIX")` an object of class `REBMIX`.

`signature(x = "REBMVNORM")` an object of class `REBMVNORM`.

Author(s)

Marko Nagode

References

H. Akaike. A new look at the statistical model identification. IEEE Transactions on Automatic Control, 19(51):716-723, 1974.

A. F. M. Smith and D. J. Spiegelhalter. Bayes factors and choice criteria for linear models. Journal of the Royal Statistical Society. Series B, 42(2):213-220, 1980. https://www.jstor.org/stable/2984964.

H. Bozdogan. Model selection and akaike's information criterion (aic): The general theory and its analytical extensions. Psychometrika, 52(3):345-370, 1987. https://doi.org/10.1007/BF02294361.

C. M. Hurvich and C.-L. Tsai. Regression and time series model selection in small samples. Biometrika, 76(2):297-307, 1989. https://www.jstor.org/stable/2336663.

AWE-methods *Approximate Weight of Evidence Criterion*

Description

Returns the approximate weight of evidence criterion at `pos`.

Usage

```
## S4 method for signature 'REBMIX'
AWE(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

`x`	see Methods section below.
`pos`	a desired row number in `x@summary` for which the information criterion is calculated. The default value is `1`.
`...`	currently not used.

Methods

`signature(x = "REBMIX")` an object of class `REBMIX`.

`signature(x = "REBMVNORM")` an object of class `REBMVNORM`.

Author(s)

Marko Nagode

References

J. D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. Biometrics, 49 (3):803-821, 1993. https://doi.org/10.2307/2532201.

`BFSMIX-methods` *Predicts Class Membership Based Upon the Best First Search Algorithm*

Description

Returns as default the optimized RCLSMIX algorithm output for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities. If `model` equals `"RCLSMVNORM"` optimized output for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices is returned.

Usage

```
## S4 method for signature 'RCLSMIX'
BFSMIX(model = "RCLSMIX", x = list(), Dataset = data.frame(),
       Zt = factor(), ...)
## ... and for other signatures
```

Arguments

`model`	see Methods section below.
`x`	a list of objects of class `REBMIX` of length o obtained by running `REBMIX` on $g = 1, \ldots, s$ train datasets $Y_{\text{train}g}$ all of length $n_{\text{train}g}$. For the train datasets the corresponding class membership $\mathbf{\Omega}_g$ is known. This yields $n_{\text{train}} = \sum_{g=1}^{s} n_{\text{train}g}$, while $Y_{\text{train}q} \cap Y_{\text{train}g} = \emptyset$ for all $q \neq g$. Each object in the list corresponds to one chunk, e.g., $(y_{1j}, y_{3j})^\top$. The default value is `list()`.
`Dataset`	a data frame containing test dataset Y_{test} of length n_{test}. For the test dataset the corresponding class membership $\mathbf{\Omega}_g$ is not known. The default value is `data.frame()`.
`Zt`	a factor of true class membership $\mathbf{\Omega}_g$ for the test dataset. The default value is `factor()`.
`...`	currently not used.

Value

Returns an optimized object of class `RCLSMIX` or `RCLSMVNORM`.

Methods

`signature(model = "RCLSMIX")` a character giving the default class name `"RCLSMIX"` for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities.

`signature(model = "RCLSMVNORM")` a character giving the class name `"RCLSMVNORM"` for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices.

Author(s)

Marko Nagode

References

R. Kohavi and G. H. John. Wrappers for feature subset selection, Artificial Intelligence, 97(1-2):273-324, 1997. https://doi.org/10.1016/S0004-3702(97)00043-X.

`BIC-methods` *Bayesian Information Criterion*

Description

Returns the Bayesian information criterion at `pos`.

Usage

```
## S4 method for signature 'REBMIX'
BIC(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

`x`	see Methods section below.
`pos`	a desired row number in `x@summary` for which the information criterion is calculated. The default value is `1`.
`...`	currently not used.

Methods

`signature(x = "REBMIX")` an object of class `REBMIX`.

`signature(x = "REBMVNORM")` an object of class `REBMVNORM`.

Author(s)

Marko Nagode

References

G. Schwarz. Estimating the dimension of the model. The Annals of Statistics, 6(2):461-464, 1978.

`boot-methods` *Parametric or Nonparametric Bootstrap for Standard Error and Coefficient of Variation Estimation*

Description

Returns as default the boot output for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities. If `x` is of class `RNGMVNORM` the boot output for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices is returned.

Usage

```
## S4 method for signature 'REBMIX'
boot(x = NULL, rseed = -1, pos = 1, Bootstrap = "parametric",
     B = 100, n = numeric(), replace = TRUE, prob = numeric(), ...)
## ... and for other signatures
## S4 method for signature 'REBMIX.boot'
summary(object, ...)
## ... and for other signatures
```

Arguments

`x`	see Methods section below.
`rseed`	set the random seed to any negative integer value to initialize the sequence. The first bootstrap dataset corresponds to it. For each next bootstrap dataset the random seed is decremented $r_{\text{seed}} = r_{\text{seed}} - 1$. The default value is `-1`.
`pos`	a desired row number in `x@summary` to be bootstrapped. The default value is `1`.
`Bootstrap`	a character giving the bootstrap type. One of default `"parametric"` or `"nonparametric"`.
`B`	number of bootstrap datasets. The default value is `100`.
`n`	number of observations. The default value is `numeric()`.
`replace`	logical. The sampling is with replacement if `TRUE`, see also `sample`. The default value is `TRUE`.
`prob`	a vector of length n containing probability weights, see also `sample`. The default value is `numeric()`.
`...`	maximum number of components `cmax`, minimum number of components `cmin` and further arguments to `sample`; additional arguments affecting the summary produced.
`object`	see Methods section below.

Value

Returns an object of class REBMIX.boot or REBMVNORM.boot.

Methods

signature(x = "REBMIX") an object of class REBMIX for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities.

signature(x = "REBMVNORM") an object of class REBMVNORM for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices.

signature(object = "REBMIX") an object of class REBMIX.

signature(object = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

G. McLachlan and D. Peel. Finite Mixture Models. John Wiley & Sons, New York, 2000.

Examples

```
data("weibull")

n <- nrow(weibull)

# Number of classes or nearest neighbours to be processed.

K <- c(as.integer(1 + log2(n)), # Minimum v follows Sturges rule.
  as.integer(10 * log10(n))) # Maximum v follows log10 rule.

# Estimate number of components, component weights and component parameters.

weibullest <- REBMIX(Dataset = list(weibull),
  Preprocessing = "kernel density estimation",
  cmax = 4,
  Criterion = "BIC",
  pdf = "Weibull",
  K = K[1]:K[2],
  Restraints = "loose")

# Plot finite mixture.

plot(weibullest, what = c("density", "distribution", "IC", "logL", "D"),
  nrow = 3, ncol = 2, npts = 1000)

# Bootstrap finite mixture.

weibullboot <- boot(x = weibullest, Bootstrap = "nonparametric", B = 10)
```

```
weibullboot
```

---

chunk-methods *Extracts Chunk from Train and Test Datasets*

Description

Returns (invisibly) the object containing train and test observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ as well as true class membership $\boldsymbol{\Omega}_g$ for the test dataset. Vectors $\boldsymbol{x}$ are subvectors of $\boldsymbol{y} = (y_1, \ldots, y_d)^\top$.

Usage

```
## S4 method for signature 'RCLS.chunk'
chunk(x = NULL, variables = expression(1:d))
## ... and for other signatures
```

Arguments

x	see Methods section below.
variables	a vector containing indices of variables in subvectors $\boldsymbol{x}$. The default value is 1:d.

Value

Returns an object of class RCLS.chunk.

Methods

signature(x = "RCLS.chunk") an object of class RCLS.chunk.

Author(s)

Marko Nagode

Examples

```
data("iris")

# Split dataset into train (75%) and test (25%) subsets.

set.seed(5)

Iris <- split(p = 0.75, Dataset = iris, class = 5)

# Extract chunk from train and test datasets.

Iris14 <- chunk(x = Iris, variables = c(1,4))

Iris14
```

CLC-methods *Classification Likelihood Criterion*

Description

Returns the classification likelihood criterion at pos.

Usage

```
## S4 method for signature 'REBMIX'
CLC(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

C. Biernacki and G. Govaert. Using the classification likelihood to choose the number of clusters. In E. J. Wegman and S. P. Azen, editors, Computing Science and Statistics, 1997.

demix-methods *Empirical Density Calculation*

Description

Returns the data frame containing observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and empirical densities $f_1, \ldots, f_n$ for the kernel density estimation or k-nearest neighbour or bin means $\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_v$ and empirical densities $f_1, \ldots, f_v$ for the histogram preprocessing. Vectors $\boldsymbol{x}$ and $\bar{\boldsymbol{x}}$ are subvectors of $\boldsymbol{y} = (y_1, \ldots, y_d)^\top$ and $\bar{\boldsymbol{y}} = (\bar{y}_1, \ldots, \bar{y}_d)^\top$.

Usage

```
## S4 method for signature 'REBMIX'
demix(x = NULL, pos = 1, variables = expression(1:d), ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the empirical densities are calculated. The default value is 1.
variables	a vector containing indices of variables in subvectors $\boldsymbol{x}$ or $\bar{\boldsymbol{x}}$. The default value is 1:d.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

M. Nagode and M. Fajdiga. The rebmix algorithm for the univariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(5):876-892, 2011a. https://doi.org/10.1080/03610920903480890.

M. Nagode and M. Fajdiga. The rebmix algorithm for the multivariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(11):2022-2034, 2011b. https://doi.org/10.1080/03610921003725788.

M. Nagode. Finite mixture modeling via REBMIX. Journal of Algorithms and Optimization, 3(2):14-28, 2015. https://doi.org/10.5963/JAO0302001.

Examples

```
# Generate simulated dataset.

n <- c(15, 15)

Theta <- new("RNGMIX.Theta", c = 2, pdf = rep("normal", 3))

a.theta1(Theta, 1) <- c(10, 20, 30)
a.theta1(Theta, 2) <- c(3, 4, 5)
a.theta2(Theta, 1) <- c(3, 2, 1)
a.theta2(Theta, 2) <- c(15, 10, 5)
```

```
simulated <- RNGMIX(Dataset.name = paste("simulated_", 1:4, sep = ""),
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))

# Number of classes or nearest neighbours to be processed.

K <- c(as.integer(1 + log2(sum(n))), # Minimum v follows Sturges rule.
  as.integer(10 * log10(sum(n)))) # Maximum v follows log10 rule.

# Estimate number of components, component weights and component parameters.

simulatedest <- REBMIX(model = "REBMVNORM",
  Dataset = a.Dataset(simulated),
  Preprocessing = "h",
  cmax = 4,
  Criterion = "BIC",
  pdf = c("n", "n", "n"),
  K = K[1]:K[2])

# Preprocess simulated dataset.

f <- demix(simulatedest, pos = 3, variables = c(1, 3))

f

# Plot finite mixture.

opar <- plot(simulatedest, pos = 3, nrow = 3, ncol = 1)

par(usr = opar[[2]]$usr, mfg = c(2, 1))

text(x = f[, 1], y = f[, 2], labels = format(f[, 3], digits = 3), cex = 0.8, pos = 1)
```

---

dfmix-methods *Predictive Marginal Density Calculation*

Description

Returns the data frame containing observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and predictive marginal densities $f(\boldsymbol{x}|c, \boldsymbol{w}, \boldsymbol{\Theta})$. Vectors $\boldsymbol{x}$ are subvectors of $\boldsymbol{y} = (y_1, \ldots, y_d)^\top$.

Usage

```
## S4 method for signature 'REBMIX'
dfmix(x = NULL, Dataset = data.frame(), pos = 1, variables = expression(1:d), ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
Dataset	a data frame containing observations $\boldsymbol{y} = (y_1, \ldots, y_d)^\top$ for which the predictive marginal densities are calculated. The default value is data.frame().
pos	a desired row number in x@summary for which the predictive marginal densities are calculated. The default value is 1.
variables	a vector containing indices of variables in subvectors $\boldsymbol{x}$. The default value is 1:d.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

M. Nagode and M. Fajdiga. The rebmix algorithm for the univariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(5):876-892, 2011a. https://doi.org/10.1080/03610920903480890.

M. Nagode and M. Fajdiga. The rebmix algorithm for the multivariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(11):2022-2034, 2011b. https://doi.org/10.1080/03610921003725788.

M. Nagode. Finite mixture modeling via REBMIX. Journal of Algorithms and Optimization, 3(2):14-28, 2015. https://doi.org/10.5963/JAO0302001.

Examples

```
# Generate simulated dataset.

n <- c(15, 15)

Theta <- new("RNGMIX.Theta", c = 2, pdf = rep("normal", 3))

a.theta1(Theta, 1) <- c(10, 20, 30)
a.theta1(Theta, 2) <- c(3, 4, 5)
a.theta2(Theta, 1) <- c(3, 2, 1)
a.theta2(Theta, 2) <- c(15, 10, 5)

simulated <- RNGMIX(Dataset.name = paste("simulated_", 1:4, sep = ""),
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))
```

```
# Number of classes or nearest neighbours to be processed.

K <- c(as.integer(1 + log2(sum(n))), # Minimum v follows Sturges rule.
  as.integer(10 * log10(sum(n)))) # Maximum v follows log10 rule.

# Estimate number of components, component weights and component parameters.

simulatedest <- REBMIX(model = "REBMVNORM",
  Dataset = a.Dataset(simulated),
  Preprocessing = "h",
  cmax = 4,
  Criterion = "BIC",
  pdf = c("n", "n", "n"))

# Preprocess simulated dataset.

Dataset <- data.frame(c(-7, 1), NA, c(3, 7))

f <- dfmix(simulatedest, Dataset = Dataset, pos = 3, variables = c(1, 3))

f

# Plot finite mixture.

opar <- plot(simulatedest, pos = 3, nrow = 3, ncol = 1,
  contour.drawlabels = TRUE, contour.labcex = 0.6)

par(usr = opar[[2]]$usr, mfg = c(2, 1))

points(x = f[, 1], y = f[, 2])

text(x = f[, 1], y = f[, 2], labels = format(f[, 3], digits = 3), cex = 0.8, pos = 4)
```

`EM.Control-class` *Class* `"EM.Control"`

Description

Object of class `EM.Control`.

Objects from the Class

Objects can be created by calls of the form `new("EM.Control", ...)`. Accessor methods for the slots are `a.strategy(x = NULL)`, `a.variant(x = NULL)`, `a.acceleration(x = NULL)`, `a.tolerance(x = NULL)`, `a.acceleration.multiplier(x = NULL)` and `a.maximum.iterations(x = NULL)` where `x` stands for an object of class `EM.Control`. Setter methods `a.strategy(x = NULL)`, `a.variant(x = NULL)`, `a.acceleration(x = NULL)`, `a.tolerance(x = NULL)`, `a.acceleration.multiplier(x = NULL)` and `a.maximum.iterations(x = NULL)` are provided to write to `strategy`, `variant`, `acceleration`, `tolerance`, `acceleration.multiplier` and `maximum.iterations` slot respectively.

Slots

`strategy`: a character containing the EM and REBMIX strategy. One of `"none"`, `"exhaustive"`, `"best"` and `"single"`. The default value is `"none"`.

`variant`: a character containing the type of the EM algorithm to be used. One of `"EM"` of `"ECM"`. The default value is `"EM"`.

`acceleration`: a character containing the type of acceleration of the EM iteration increment. One of `"fixed"`, `"line"` or `"golden"`. The default value is `"fixed"`.

`tolerance`: tolerance value for the EM convergence criteria. The default value is 1e-4.

`acceleration.multiplier`: acceleration.multiplier a_{EM}, $1.0 \leq a_{\mathrm{EM}} \leq 2.0$. acceleration.multiplier for the EM step increment. The default value is 1.0.

`maximum.iterations`: maximum.iterations is positive integer containing the maximum allowed number of iterations of the EM algorithm. The default value is 1000.

Author(s)

Panic Branislav

References

A. P. Dempster et al. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B, 39(1):1-38, 1977. https://www.jstor.org/stable/2984875.

G. Celeux and G. Govaert. A classification EM algorithm for clustering and two stochastic versions, Computational Statistics & Data Analysis, 14(3):315:332, 1992. https://doi.org/10.1016/0167-9473(92)90042-E.

Examples

```
# Inline creation by function new call.

EM <- new("EM.Control", strategy = "exhaustive",
  variant = "EM", acceleration = "fixed",
  tolerance = 1e-4, acceleration.multiplier = 1.0,
  maximum.iterations = 1000)

EM

# Creation of EM object with setter functions.

EM <- new("EM.Control")

a.strategy(EM) <- "exhaustive"
a.variant(EM) <- "EM"
a.acceleration(EM) <- "fixed"
a.tolerance(EM) <- 1e-4
a.acceleration.multiplier(EM) <- 1.0
a.maximum.iterations(EM) <- 1000
```

EM

---

galaxy *Galaxy Dataset*

Description

The unfilled survey of the Corona Borealis region contains the velocities of 82 galaxies from 6 well separated conic sections of space.

Usage

```
data("galaxy")
```

Format

galaxy is a data frame with 82 cases (rows) and 1 continuous variable (columns) called Velocity.

Source

K. Roeder. Density estimation with confidence sets exemplified by superclusters and voids in the galaxies. Journal of American Statistical Association, 85(411):617-624, 1990. https://www.jstor.org/stable/2289993.

References

S. Richardson and P. J. Green. On bayesian analysis of mixtures with an unknown number of components. Journal of the Royal Statistical Society B, 59(4):731-792, 1997. https://www.jstor.org/stable/2985194.

G. McLachlan and D. Peel. Contribution to the discussion of paper by s. richardson and p.j. green. Journal of the Royal Statistical Society B, 59(4):779-780, 1997. https://www.jstor.org/stable/2985194.

M. Stephens. Bayesian analysis of mixture models with an unknown number of components - an alternative to reversible jump methods. The Annals of Statistics, 28(1):40-74, 2000. https://www.jstor.org/stable/2673981.

HQC-methods *Hannan-Quinn Information Criterion*

Description

Returns the Hannan-Quinn information criterion at pos.

Usage

```
## S4 method for signature 'REBMIX'
HQC(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

E. J. Hannan and B. G. Quinn. The determination of the order of an autoregression. Journal of the Royal Statistical Society. Series B, 41(2):190-195, 1979. https://www.jstor.org/stable/2985032.

ICL-methods *Integrated Classification Likelihood Criterion*

Description

Returns the integrated classification likelihood criterion at pos.

Usage

```
## S4 method for signature 'REBMIX'
ICL(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated classification likelihood. Technical Report 3521, INRIA, Rhone-Alpes, 1998.

ICLBIC-methods *Approximate Integrated Classification Likelihood Criterion*

Description

Returns the approximate integrated classification likelihood criterion at pos.

Usage

```
## S4 method for signature 'REBMIX'
ICLBIC(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

`signature(x = "REBMIX")` an object of class `REBMIX`.

`signature(x = "REBMVNORM")` an object of class `REBMVNORM`.

Author(s)

Marko Nagode

References

C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated classification likelihood. Technical Report 3521, INRIA, Rhone-Alpes, 1998.

`iris` *Iris Data Set*

Description

This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

Usage

```
data("iris")
```

Format

`iris` is a data frame with 150 cases (rows) and 5 variables (columns) named:

1. `Sepal.Length` continuous.
2. `Sepal.Width` continuous.
3. `Petal.Length` continuous.
4. `Petal.Width` continuous.
5. `Class` discrete `iris-setosa`, `iris-versicolour` or `iris-virginica`.

Source

A. Asuncion and D. J. Newman. Uci machine learning repository, 2007. http://archive.ics.uci.edu/ml.

References

R. A. Fisher. The use of multiple measurements in taxonomic problems. Annals of Eugenics, 7(2):179-188, 1936.

Examples

```
## Not run:
devAskNewPage(ask = TRUE)

data("iris")

# Show level attributes.

levels(iris[["Class"]])

# Split dataset into train (75

set.seed(5)

Iris <- split(p = 0.75, Dataset = iris, class = 5)

# Estimate number of components, component weights and component
# parameters for train subsets.

n <- range(a.ntrain(Iris))

irisest <- REBMIX(model = "REBMVNORM",
  Dataset = a.train(Iris),
  Preprocessing = "kernel density estimation",
  cmax = 10,
  Criterion = "ICL-BIC",
  pdf = rep("normal", 4))

plot(irisest, pos = 1, nrow = 3, ncol = 2, what = c("den"))
plot(irisest, pos = 2, nrow = 3, ncol = 2, what = c("den"))
plot(irisest, pos = 3, nrow = 3, ncol = 2, what = c("den"))

# Selected chunks.

iriscla <- RCLSMIX(model = "RCLSMVNORM",
  x = list(irisest),
  Dataset = a.test(Iris),
  Zt = a.Zt(Iris))

iriscla

summary(iriscla)

# Plot selected chunks.

plot(iriscla, nrow = 3, ncol = 2)

## End(Not run)
```

---

kseq *Sequence of Bins or Nearest Neighbours Generation*

Description

Returns (invisibly) a vector containing numbers of bins v for the histogram and the kernel density estimation or numbers of nearest neighbours k for the k-nearest neighbour.

Usage

```
kseq(from = NULL, to = NULL, f = 0.05, ...)
```

Arguments

from	starting value of the sequence. The default value is NULL.
to	end value of the sequence. The default value is NULL.
f	number specifying the fraction by which the bins or nearest neighbours should be separated $0.0 < f < 1.0$. The default value is 0.05.
...	currently not used.

Author(s)

Marko Nagode

Examples

```
# Generate numbers of bins.

n <- 10000

Sturges <- as.integer(1 + log2(n)) # Minimum v follows Sturges rule.
Log10 <- as.integer(10 * log10(n)) # Maximum v follows Log10 rule.
RootN <- as.integer(2 * n^0.5) # Maximum v follows RootN rule.

K <- kseq(from = Sturges, to = Log10, f = 0.05)

K

K <- kseq(from = Sturges, to = RootN, f = 0.03)

K
```

---

logL *Log Likelihood*

Description

Returns the log likelihood at pos.

Usage

```
## S4 method for signature 'REBMIX'
logL(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

G. McLachlan and D. Peel. Finite Mixture Models. John Wiley & Sons, New York, 2000.

MDL-methods *Minimum Description Length*

Description

Returns the minimum desription length at pos.

Usage

```
## S4 method for signature 'REBMIX'
MDL2(x = NULL, pos = 1, ...)
## S4 method for signature 'REBMIX'
MDL5(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

`signature(x = "REBMIX")` an object of class `REBMIX`.

`signature(x = "REBMVNORM")` an object of class `REBMVNORM`.

Author(s)

Marko Nagode

References

M. H. Hansen and B. Yu. Model selection and the principle of minimum description length. Journal of the American Statistical Association, 96(454):746-774, 2001. https://www.jstor.org/stable/2670311.

PC-methods *Partition Coefficient*

Description

Returns the partition coefficient of Bezdek at `pos`.

Usage

```
## S4 method for signature 'REBMIX'
PC(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

`x`	see Methods section below.
`pos`	a desired row number in `x@summary` for which the information criterion is calculated. The default value is `1`.
`...`	currently not used.

Methods

`signature(x = "REBMIX")` an object of class `REBMIX`.

`signature(x = "REBMVNORM")` an object of class `REBMVNORM`.

Author(s)

Marko Nagode

References

G. McLachlan and D. Peel. Finite Mixture Models. John Wiley & Sons, New York, 2000.

pemix-methods *Empirical Distribution Function Calculation*

Description

Returns the data frame containing observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and empirical distribution functions $F_1, \ldots, F_n$. Vectors $\boldsymbol{x}$ are subvectors of $\boldsymbol{y} = (y_1, \ldots, y_d)^\top$.

Usage

```
## S4 method for signature 'REBMIX'
pemix(x = NULL, pos = 1, variables = expression(1:d),
      lower.tail = TRUE, log.p = FALSE, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the empirical distribution functions are calculated. The default value is 1.
variables	a vector containing indices of variables in subvectors $\boldsymbol{x}$. The default value is 1:d.
lower.tail	logical. If TRUE, probabilities are $P[X \leq x]$, otherwise, $P[X > x]$. The default value is TRUE.
log.p	logical. if TRUE, probabilities p are given as $\log(p)$. The default value is FALSE.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

M. Nagode and M. Fajdiga. The rebmix algorithm for the univariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(5):876-892, 2011a. https://doi.org/10.1080/03610920903480890.

M. Nagode and M. Fajdiga. The rebmix algorithm for the multivariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(11):2022-2034, 2011b. https://doi.org/10.1080/03610921003725788.

M. Nagode. Finite mixture modeling via REBMIX. Journal of Algorithms and Optimization, 3(2):14-28, 2015. https://doi.org/10.5963/JAO0302001.

Examples

```
# Generate simulated dataset.

n <- c(15, 15)

Theta <- new("RNGMIX.Theta", c = 2, pdf = rep("normal", 3))

a.theta1(Theta, 1) <- c(10, 20, 30)
a.theta1(Theta, 2) <- c(3, 4, 5)
a.theta2(Theta, 1) <- c(3, 2, 1)
a.theta2(Theta, 2) <- c(15, 10, 5)

simulated <- RNGMIX(Dataset.name = paste("simulated_", 1:4, sep = ""),
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))

# Number of classes or nearest neighbours to be processed.

K <- c(as.integer(1 + log2(sum(n))), # Minimum v follows Sturges rule.
  as.integer(10 * log10(sum(n)))) # Maximum v follows log10 rule.

# Estimate number of components, component weights and component parameters.

simulatedest <- REBMIX(Dataset = a.Dataset(simulated),
  Preprocessing = "kernel density estimation",
  cmax = 4,
  Criterion = "BIC",
  pdf = c("n", "n", "n"),
  K = K[1]:K[2])

# Preprocess simulated dataset.

f <- pemix(simulatedest, pos = 3, variables = c(1, 2))

f

# Plot finite mixture.

opar <- plot(simulatedest, pos = 3, nrow = 3, ncol = 1, what = "dist")

par(usr = opar[[1]]$usr, mfg = c(1, 1))

text(x = f[20:25, 1], y = f[20:25, 2], labels = format(f[20:25, 3],
  digits = 3), cex = 0.8, pos = 1)
```

pfmix-methods *Predictive Marginal Distribution Function Calculation*

Description

Returns the data frame containing observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and predictive marginal distribution functions $F(\boldsymbol{x}|c, \boldsymbol{w}, \boldsymbol{\Theta})$. Vectors $\boldsymbol{x}$ are subvectors of $\boldsymbol{y} = (y_1, \ldots, y_d)^\top$.

Usage

```
## S4 method for signature 'REBMIX'
pfmix(x = NULL, Dataset = data.frame(), pos = 1,
      variables = expression(1:d), lower.tail = TRUE, log.p = FALSE, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
Dataset	a data frame containing observations $\boldsymbol{y} = (y_1, \ldots, y_d)^\top$ for which the predictive marginal distribution functions are calculated. The default value is data.frame().
pos	a desired row number in x@summary for which the predictive marginal distribution functions are calculated. The default value is 1.
variables	a vector containing indices of variables in subvectors $\boldsymbol{x}$. The default value is 1:d.
lower.tail	logical. If TRUE, probabilities are $P[X \leq x]$, otherwise, $P[X > x]$. The default value is TRUE.
log.p	logical. if TRUE, probabilities p are given as $\log(p)$. The default value is FALSE.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

M. Nagode and M. Fajdiga. The rebmix algorithm for the univariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(5):876-892, 2011a. https://doi.org/10.1080/03610920903480890.

M. Nagode and M. Fajdiga. The rebmix algorithm for the multivariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(11):2022-2034, 2011b. https:

//doi.org/10.1080/03610921003725788.

M. Nagode. Finite mixture modeling via REBMIX. Journal of Algorithms and Optimization, 3(2):14-28, 2015. https://doi.org/10.5963/JAO0302001.

Examples

```
# Generate simulated dataset.

n <- c(15, 15)

Theta <- new("RNGMIX.Theta", c = 2, pdf = rep("normal", 3))

a.theta1(Theta, 1) <- c(10, 20, 30)
a.theta1(Theta, 2) <- c(3, 4, 5)
a.theta2(Theta, 1) <- c(3, 2, 1)
a.theta2(Theta, 2) <- c(15, 10, 5)

simulated <- RNGMIX(Dataset.name = paste("simulated_", 1:4, sep = ""),
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))

# Number of classes or nearest neighbours to be processed.

K <- c(as.integer(1 + log2(sum(n))), # Minimum v follows Sturges rule.
  as.integer(10 * log10(sum(n)))) # Maximum v follows log10 rule.

# Estimate number of components, component weights and component parameters.

simulatedest <- REBMIX(Dataset = a.Dataset(simulated),
  Preprocessing = "h",
  cmax = 4,
  Criterion = "BIC",
  pdf = c("n", "n", "n"))

# Preprocess simulated dataset.

Dataset <- data.frame(c(25, 5, -20), NA, c(31, 20, 20))

f <- pfmix(simulatedest, Dataset = Dataset, pos = 3, variables = c(1, 3))

f

# Plot finite mixture.

opar <- plot(simulatedest, pos = 3, nrow = 3, ncol = 1,
  what = "dist", contour.drawlabels = TRUE, contour.labcex = 0.6)

par(usr = opar[[2]]$usr, mfg = c(2, 1))

points(x = f[, 1], y = f[, 2])
```

```
text(x = f[, 1], y = f[, 2], labels = format(f[, 3], digits = 3), cex = 0.8, pos = 4)
```

---

plot-methods *Plots RNGMIX, REBMIX, RCLRMIX and RCLSMIX Output*

Description

Plots true clusters if x equals "RNGMIX". Plots the REBMIX output depending on what argument if x equals "REBMIX". Plots predictive clusters if x equals "RCLRMIX". Wrongly clustered observations are plotted only if x@Zt is available. Plots predictive classes and wrongly classified observations if x equals "RCLSMIX".

Usage

```
## S4 method for signature 'RNGMIX,missing'
plot(x, y, pos = 1, nrow = 1, ncol = 1, cex = 0.8,
     fg = "black", lty = "solid", lwd = 1, pty = "m", tcl = 0.5,
     plot.cex = 0.8, plot.pch = 19, ...)
## S4 method for signature 'REBMIX,missing'
plot(x, y, pos = 1, what = c("density"),
     nrow = 1, ncol = 1, npts = 200, n = 200, cex = 0.8, fg = "black",
     lty = "solid", lwd = 1, pty = "m", tcl = 0.5,
     plot.cex = 0.8, plot.pch = 19, contour.drawlabels = FALSE,
     contour.labcex = 0.8, contour.method = "flattest",
     contour.nlevels = 12, ...)
## S4 method for signature 'RCLRMIX,missing'
plot(x, y, s = expression(c), nrow = 1, ncol = 1, cex = 0.8,
     fg = "black", lty = "solid", lwd = 1, pty = "m", tcl = 0.5,
     plot.cex = 0.8, plot.pch = 19, ...)
## S4 method for signature 'RCLSMIX,missing'
plot(x, y, nrow = 1, ncol = 1, cex = 0.8,
     fg = "black", lty = "solid", lwd = 1, pty = "m", tcl = 0.5,
     plot.cex = 0.8, plot.pch = 19, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
y	currently not used.
pos	a desired row number in x@summary or a desired element number in x@Dataset to be plotted. The default value is 1.
s	a desired number of clusters to be plotted. The default value is expression(c).
what	a character vector giving the plot types. One of "density" for probability density function, "marginal" for marginal probability density function, "IC" for information criterion depending on numbers of components c, "logL" for log

likelihood, "D" for total of positive relative deviations, "distribution" for distribution function or "K" for information criterion depending on bins v or numbers of nearest neighbours k. The default value is "density".

nrow — a desired number of rows in which the empirical and predictive densities are to be plotted. The default value is 1.

ncol — a desired number of columns in which the empirical and predictive densities are to be plotted. The default value is 1.

npts — a number of points at which the predictive densities are to be plotted. The default value is 200.

n — a number of observations to be plotted. The default value is 200.

cex — a numerical value giving the amount by which the plotting text and symbols should be magnified relative to the default, see also par. The default value is 0.8.

fg — a colour used for things like axes and boxes around plots, see also par. The default value is "black".

lty — a line type, see also par. The default value is "solid".

lwd — a line width, see also par. The default value is 1.

pty — a character specifying the type of the plot region to be used. One of "s" generating a square plotting region or "m" generating the maximal plotting region. The default value is "m".

tcl — a length of tick marks as a fraction of the height of a line of the text, see also par. The default value is 0.5.

plot.cex — a numerical vector giving the amount by which plotting characters and symbols should be scaled relative to the default. It works as a multiple of par("cex"). NULL and NA are equivalent to 1.0. Note that this does not affect annotation, see also plot.default. The default value is 0.8.

plot.pch — a vector of plotting characters or symbols, see also points. The default value is 19.

contour.drawlabels — logical. The contours are labelled if TRUE. The default value is FALSE.

contour.labcex — cex for contour labelling. The default value is 0.8. This is an absolute size, not a multiple of par("cex").

contour.method — a character specifying where the labels will be located. The possible values are "simple", "edge" and default "flattest", see also contour.

contour.nlevels — a number of desired contour levels. The default value is 12.

... — further arguments to par.

Value

Returns (invisibly) a list containing graphical parameters par. Such a list can be passed as an argument to par to restore the parameter values.

Methods

`signature(x = "RNGMIX", y = "missing")` an object of class RNGMIX.

`signature(x = "RNGMVNORM", y = "missing")` an object of class RNGMVNORM.

`signature(x = "REBMIX", y = "missing")` an object of class REBMIX.

`signature(x = "REBMVNORM", y = "missing")` an object of class REBMVNORM.

`signature(x = "RCLRMIX", y = "missing")` an object of class RCLRMIX.

`signature(x = "RCLRMVNORM", y = "missing")` an object of class RCLRMVNORM.

`signature(x = "RCLSMIX", y = "missing")` an object of class RCLSMIX.

`signature(x = "RCLSMVNORM", y = "missing")` an object of class RCLSMVNORM.

Author(s)

Marko Nagode

References

C. M. Bishop. Neural Networks for Pattern Recognition. Clarendon Press, Oxford, 1995.

Examples

```
## Not run:
devAskNewPage(ask = TRUE)

data("wine")

colnames(wine)

# Remove Cultivar column from wine dataset.

winecolnames <- !(colnames(wine) %in% "Cultivar")

wine <- wine[, winecolnames]

# Determine number of dimensions d and wine dataset size n.

d <- ncol(wine)
n <- nrow(wine)

# Estimate number of components, component weights and component parameters.

Sturges <- as.integer(1 + log2(n)) # Minimum v follows Sturges rule.
RootN <- as.integer(2 * n^0.5) # Maximum v follows RootN rule.

K <- c(floor(Sturges^(1/13)), ceiling(RootN^(1/13)))

wineest <- REBMIX(model = "REBMVNORM",
  Dataset = list(wine = wine),
  Preprocessing = "kernel density estimation",
  Criterion = "ICL-BIC",
```

```
  pdf = rep("normal", d),
  K = K[1]:K[2])

# Plot finite mixture.

plot(wineest, what = c("density", "IC", "logL", "D"),
  nrow = 2, ncol = 2, pty = "s")

## End(Not run)
```

PRD-methods *Total of Positive Relative Deviations*

Description

Returns the total of positive relative deviations D at pos.

Usage

```
## S4 method for signature 'REBMIX'
PRD(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

M. Nagode and M. Fajdiga. The rebmix algorithm for the univariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(5):876-892, 2011a. https://doi.org/10.1080/03610920903480890.

M. Nagode and M. Fajdiga. The rebmix algorithm for the multivariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(11):2022-2034, 2011b. https://doi.org/10.1080/03610921003725788.

M. Nagode. Finite mixture modeling via REBMIX. Journal of Algorithms and Optimization, 3(2):14-28, 2015. https://doi.org/10.5963/JAO0302001.

RCLRMIX-class *Class* "RCLRMIX"

Description

Object of class RCLRMIX.

Objects from the Class

Objects can be created by calls of the form new("RCLRMIX",...). Accessor methods for the slots are a.pos(x = NULL), a.Zt(x = NULL), a.Zp(x = NULL,s = expression(c)), a.c(x = NULL), a.p(x = NULL,s = expression(c)), a.pi(x = NULL,s = expression(c)), a.P(x = NULL,s = expression(c)), a.tau(x = NULL,s = expression(c)), a.prob(x = NULL), a.from(x = NULL), a.to(x = NULL), a.EN(x = NULL) and a.ED(x = NULL), where x stands for an object of class RCLRMIX and s a desired number of clusters for which the slot is calculated.

Slots

x: an object of class REBMIX.

pos: a desired row number in x@summary to be clustered. The default value is 1.

Zt: a factor of true cluster membership.

Zp: a factor of predictive cluster membership.

c: number of clusters.

p: a vector of length c containing prior probabilities of cluster memberships p_l summing to 1. The value is returned only if all variables in slot x follow either binomial or Dirac parametric families. The default value is numeric().

pi: a list of length d of matrices of size $c \times K_i$ containing cluster conditional probabilities π_{ilk}. Let π_{ilk} denote the cluster conditional probability that an observation in cluster $l = 1, \ldots, c$ produces the kth outcome on the ith variable. Suppose we observe $i = 1, \ldots, d$ polytomous categorical variables (the manifest variables), each of which contains K_i possible outcomes for observations $j = 1, \ldots, n$. A manifest variable is a variable that can be measured or observed directly. It must be coded as whole number starting at zero for the first outcome and increasing to the possible number of outcomes minus one. It is presumed here that all variables are statistically independentand within clusters and that $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ stands for an observed d dimensional dataset of size n of vector observations $\boldsymbol{y}_j = (y_{1j}, \ldots, y_{ij}, \ldots, y_{dj})^\top$. The value is returned only if all variables in slot x follow either binomial or Dirac parametric families. The default value is list().

P: a data frame containing true $N_\mathrm{t}(\boldsymbol{y}_{\tilde{j}})$ and predictive $N_\mathrm{p}(\boldsymbol{y}_{\tilde{j}})$ frequencies calculated for unique $\boldsymbol{y}_{\tilde{j}} \in \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\}$, where $\tilde{j} = 1, \ldots, \tilde{n}$ and $\tilde{n} \leq n$.

tau: a matrix of size $n \times c$ containing conditional probabilities τ_{jl} that observations $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ arise from clusters $1, \ldots, c$.

`prob`: a vector of length c containing probabilities of correct clustering for $s = 1, \ldots, c$.

`from`: a vector of length $c - 1$ containing clusters merged to `to` clusters.

`to`: a vector of length $c - 1$ containing clusters originating from `from` clusters.

`EN`: a vector of length $c - 1$ containing entropies for combined clusters.

`ED`: a vector of length $c - 1$ containing decrease of entropies for combined clusters.

Author(s)

Marko Nagode

References

J. P. Baudry, A. E. Raftery, G. Celeux, K. Lo and R. Gottardo. Combining mixture components for clustering. Journal of Computational and Graphical Statistics, 19(2):332-353, 2010. https://doi.org/10.1198/jcgs.2010.08111

Examples

```
devAskNewPage(ask = TRUE)

# Generate normal dataset.

n <- c(500, 200, 400)

Theta <- new("RNGMVNORM.Theta", c = 3, d = 2)

a.theta1(Theta, 1) <- c(3, 10)
a.theta1(Theta, 2) <- c(8, 6)
a.theta1(Theta, 3) <- c(12, 11)
a.theta2(Theta, 1) <- c(3, 0.3, 0.3, 2)
a.theta2(Theta, 2) <- c(5.7, -2.3, -2.3, 3.5)
a.theta2(Theta, 3) <- c(2, 1, 1, 2)

normal <- RNGMIX(model = "RNGMVNORM", Dataset.name = "normal_1", n = n, Theta = a.Theta(Theta))

# Estimate number of components, component weights and component parameters.

normalest <- REBMIX(model = "REBMVNORM",
  Dataset = a.Dataset(normal),
  Preprocessing = "histogram",
  cmax = 6,
  Criterion = "BIC",
  pdf = rep("normal", 2))

summary(normalest)

# Plot finite mixture.

plot(normalest)

# Cluster dataset.
```

```
normalclu <- RCLRMIX(model = "RCLRMVNORM", x = normalest, Zt = a.Zt(normal))

# Plot clusters.

plot(normalclu)

summary(normalclu)
```

RCLRMIX-methods *Predicts Cluster Membership Based Upon a Model Trained by REBMIX*

Description

Returns as default the RCLRMIX algorithm output for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities, following the methodology proposed in the article cited in the references. If `model` equals `"RCLRMVNORM"` output for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices is returned.

Usage

```
## S4 method for signature 'RCLRMIX'
RCLRMIX(model = "RCLRMIX", x = NULL, pos = 1, Zt = factor(), ...)
## ... and for other signatures
## S4 method for signature 'RCLRMIX'
summary(object, ...)
## ... and for other signatures
```

Arguments

`model`	see Methods section below.
`x`	an object of class `REBMIX`.
`pos`	a desired row number in `x@summary` to be clustered. The default value is `1`.
`Zt`	a factor of true cluster membership. The default value is `factor()`.
`object`	see Methods section below.
`...`	currently not used; additional arguments affecting the summary produced.

Value

Returns an object of class `RCLRMIX` or `RCLRMVNORM`.

Methods

`signature(model = "RCLRMIX")` a character giving the default class name `"RCLRMIX"` for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities.

`signature(model = "RCLRMVNORM")` a character giving the class name `"RCLRMVNORM"` for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices.

`signature(object = "RCLRMIX")` an object of class `RCLRMIX`.

`signature(object = "RCLRMVNORM")` an object of class `RCLRMVNORM`.

Author(s)

Marko Nagode

References

J. P. Baudry, A. E. Raftery, G. Celeux, K. Lo and R. Gottardo. Combining mixture components for clustering. Journal of Computational and Graphical Statistics, 19(2):332-353, 2010. https://doi.org/10.1198/jcgs.2010.08111

Examples

```
devAskNewPage(ask = TRUE)

# Generate Poisson dataset.

n <- c(500, 200, 400)

Theta <- new("RNGMIX.Theta", c = 3, pdf = "Poisson")

a.theta1(Theta) <- c(3, 12, 36)

poisson <- RNGMIX(Dataset.name = "Poisson_1", n = n, Theta = a.Theta(Theta))

# Estimate number of components, component weights and component parameters.

poissonest <- REBMIX(Dataset = a.Dataset(poisson),
  Preprocessing = "histogram",
  cmax = 6,
  Criterion = "BIC",
  pdf = rep("Poisson", 1),
  K = 1)

summary(poissonest)

# Plot finite mixture.

plot(poissonest)

# Cluster dataset.
```

```
poissonclu <- RCLRMIX(x = poissonest, Zt = a.Zt(poisson))

summary(poissonclu)

# Plot clusters.

plot(poissonclu)

summary(poissonclu)
```

RCLS.chunk-class *Class* "RCLS.chunk"

Description

Object of class RCLS.chunk.

Objects from the Class

Objects can be created by calls of the form new("RCLS.chunk",...). Accessor methods for the slots are a.s(x = NULL), a.levels(x = NULL), a.ntrain(x = NULL), a.train(x = NULL), a.Zr(x = NULL), a.ntest(x = NULL), a.test(x = NULL) and a.Zt(x = NULL), where x stands for an object of class RCLS.chunk.

Slots

s: finite set of size s of classes $\mathbf{\Omega} = \{\mathbf{\Omega}_g;\ g = 1, \ldots, s\}$.

levels: a character vector of length s containing class names $\mathbf{\Omega}_g$.

ntrain: a vector of length s containing numbers of observations in train datasets $Y_{\text{train}g}$.

train: a list of data frames containing train datasets $Y_{\text{train}g}$ of length $n_{\text{train}g}$.

Zr: a list of factors of true class membership $\mathbf{\Omega}_g$ for the train datasets.

ntest: number of observations in test dataset Y_{test}.

test: a data frame containing test dataset Y_{test} of length n_{test}.

Zt: a factor of true class membership $\mathbf{\Omega}_g$ for the test dataset.

Author(s)

Marko Nagode

References

D. M. Dziuda. Data Mining for Genomics and Proteomics: Analysis of Gene and Protein Expression Data. John Wiley & Sons, New York, 2010.

RCLSMIX-class *Class* "RCLSMIX"

Description

Object of class RCLSMIX.

Objects from the Class

Objects can be created by calls of the form new("RCLSMIX",...). Accessor methods for the slots are a.o(x = NULL), a.Dataset(x = NULL), a.s(x = NULL), a.ntrain(x = NULL), a.P(x = NULL), a.ntest(x = NULL), a.Zt(x = NULL), a.Zp(x = NULL), a.CM(x = NULL), a.Accuracy(x = NULL), a.Error(x = NULL), a.Precision(x = NULL), a.Sensitivity(x = NULL), a.Specificity(x = NULL) and a.Chunks(x = NULL), where x stands for an object of class RCLSMIX.

Slots

x: a list of objects of class REBMIX of length o obtained by running REBMIX on $g = 1, \ldots, s$ train datasets $Y_{\text{train}g}$ all of length $n_{\text{train}g}$. For the train datasets the corresponding class membership $\boldsymbol{\Omega}_g$ is known. This yields $n_{\text{train}} = \sum_{g=1}^{s} n_{\text{train}g}$, while $Y_{\text{train}q} \cap Y_{\text{train}g} = \emptyset$ for all $q \neq g$. Each object in the list corresponds to one chunk, e.g., $(y_{1j}, y_{3j})^\top$.

o: number of chunks o. $Y = \{\boldsymbol{y}_j;\ j = 1, \ldots, n\}$ is an observed d-dimensional dataset of size n of vector observations $\boldsymbol{y}_j = (y_{1j}, \ldots, y_{dj})^\top$ and is partitioned into train and test datasets. Vector observations $\boldsymbol{y}_j$ may further be split into o chunks when running REBMIX, e.g., for $d = 6$ and $o = 3$ the set of chunks substituting $\boldsymbol{y}_j$ may be as follows $(y_{1j}, y_{3j})^\top$, $(y_{2j}, y_{4j}, y_{6j})^\top$ and y_{5j}.

Dataset: a data frame containing test dataset Y_{test} of length n_{test}. For the test dataset the corresponding class membership $\boldsymbol{\Omega}_g$ is not known.

s: finite set of size s of classes $\boldsymbol{\Omega} = \{\boldsymbol{\Omega}_g;\ g = 1, \ldots, s\}$.

ntrain: a vector of length s containing numbers of observations in train datasets $Y_{\text{train}g}$.

P: a vector of length s containing prior probabilities $P(\boldsymbol{\Omega}_g) = \frac{n_{\text{train}g}}{n_{\text{train}}}$.

ntest: number of observations in test dataset Y_{test}.

Zt: a factor of true class membership $\boldsymbol{\Omega}_g$ for the test dataset.

Zp: a factor of predictive class membership $\boldsymbol{\Omega}_g$ for the test dataset.

CM: a table containing confusion matrix for multiclass classifier. It contains number x_{qg} of test observations with the true class q that are classified into the class g, where $q, g = 1, \ldots, s$.

Accuracy: proportion of all test observations that are classified correctly. $\text{Accuracy} = \frac{\sum_{g=1}^{s} x_{gg}}{n_{\text{test}}}$.

Error: proportion of all test observations that are classified wrongly. $\text{Error} = 1 - \text{Accuracy}$.

Precision: a vector containing proportions of predictive observations in class g that are classified correctly into class g. $\text{Precision}(g) = \frac{x_{gg}}{\sum_{q=1}^{s} x_{qg}}$.

Sensitivity: a vector containing proportions of test observations in class g that are classified correctly into class g. $\text{Sensitivity}(g) = \frac{x_{gg}}{\sum_{q=1}^{s} x_{gq}}$.

`Specificity`: a vector containing proportions of test observations that are not in class g and are classified into the non g class. $\text{Specificity}(g) = \frac{n_{\text{test}} - \sum_{q=1}^{s} x_{qg}}{n_{\text{test}} - \sum_{q=1}^{s} x_{gq}}$.

`Chunks`: a vector containing selected chunks.

Author(s)

Marko Nagode

References

D. M. Dziuda. Data Mining for Genomics and Proteomics: Analysis of Gene and Protein Expression Data. John Wiley & Sons, New York, 2010.

`RCLSMIX-methods` *Predicts Class Membership Based Upon a Model Trained by REBMIX*

Description

Returns as default the RCLSMIX algorithm output for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities. If `model` equals `"RCLSMVNORM"` output for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices is returned.

Usage

```
## S4 method for signature 'RCLSMIX'
RCLSMIX(model = "RCLSMIX", x = list(), Dataset = data.frame(),
        Zt = factor(), ...)
## ... and for other signatures
## S4 method for signature 'RCLSMIX'
summary(object, ...)
## ... and for other signatures
```

Arguments

`model`	see Methods section below.
`x`	a list of objects of class `REBMIX` of length o obtained by running `REBMIX` on $g = 1, \ldots, s$ train datasets $Y_{\text{train}g}$ all of length $n_{\text{train}g}$. For the train datasets the corresponding class membership $\mathbf{\Omega}_g$ is known. This yields $n_{\text{train}} = \sum_{g=1}^{s} n_{\text{train}g}$, while $Y_{\text{train}q} \cap Y_{\text{train}g} = \emptyset$ for all $q \neq g$. Each object in the list corresponds to one chunk, e.g., $(y_{1j}, y_{3j})^{\top}$. The default value is `list()`.
`Dataset`	a data frame containing test dataset Y_{test} of length n_{test}. For the test dataset the corresponding class membership $\mathbf{\Omega}_g$ is not known. The default value is `data.frame()`.
`Zt`	a factor of true class membership $\mathbf{\Omega}_g$ for the test dataset. The default value is `factor()`.

object	see Methods section below.
...	currently not used; additional arguments affecting the summary produced.

Value

Returns an object of class RCLSMIX or RCLSMVNORM.

Methods

signature(model = "RCLSMIX") a character giving the default class name "RCLSMIX" for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities.

signature(model = "RCLSMVNORM") a character giving the class name "RCLSMVNORM" for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices.

signature(object = "RCLSMIX") an object of class RCLSMIX.

signature(object = "RCLSMVNORM") an object of class RCLSMVNORM.

Author(s)

Marko Nagode

References

R. O. Duda and P. E. Hart. Pattern Classification and Scene Analysis. John Wiley & Sons, New York, 1973.

Examples

```
## Not run:
devAskNewPage(ask = TRUE)

data("adult")

# Find complete cases.

adult <- adult[complete.cases(adult),]

# Replace levels with numbers.

adult <- as.data.frame(data.matrix(adult))

# Find numbers of levels.

cmax <- unlist(lapply(apply(adult[, c(-1, -16)], 2, unique), length))

cmax

# Split adult dataset into train and test subsets for two Incomes
# and remove Type and Income columns.
```

```
Adult <- split(p = list(type = 1, train = 2, test = 1),
  Dataset = adult, class = 16)

# Estimate number of components, component weights and component parameters
# for the set of chunks 1:14.

adultest <- list()

for (i in 1:14) {
  adultest[[i]] <- REBMIX(Dataset = a.train(chunk(Adult, i)),
    Preprocessing = "histogram",
    cmax = min(120, cmax[i]),
    Criterion = "BIC",
    pdf = "Dirac",
    K = 1)
}

# Class membership prediction based upon the best first search algorithm.

adultcla <- BFSMIX(x = adultest,
  Dataset = a.test(Adult),
  Zt = a.Zt(Adult))

adultcla

summary(adultcla)

# Plot selected chunks.

plot(adultcla, nrow = 5, ncol = 2)

## End(Not run)
```

REBMIX-class *Class* "REBMIX"

Description

Object of class REBMIX.

Objects from the Class

Objects can be created by calls of the form new("REBMIX",...). Accessor methods for the slots are a.Dataset(x = NULL,pos = 0), a.Preprocessing(x = NULL), a.cmax(x = NULL), a.cmin(x = NULL), a.Criterion(x = NULL), a.Variables(x = NULL), a.pdf(x = NULL), a.theta1(x = NULL), a.theta2(x = NULL), a.K(x = NULL), a.y0(x = NULL), a.ymin(x = NULL), a.ymax(x = NULL), a.ar(x = NULL), a.Restraints(x = NULL), a.w(x = NULL,pos = 0), a.Theta(x = NULL,pos = 0), a.summary(x = NULL,pos = 0,col.name = character()), a.summary.EM(x = NULL,pos = 0,col.name = character()), a.pos(x = NULL), a.opt.c(x = NULL), a.opt.IC(x = NULL), a.opt.logL(x =

NULL), a.opt.D(x = NULL), a.all.K(x = NULL), a.all.IC(x = NULL), a.theta1.all(x = NULL,pos = 1) and a.theta2.all(x = NULL,pos = 1), where x, pos and col.name stand for an object of class REBMIX, a desired slot item and a desired column name, respectively.

Slots

Dataset: a list of data frames of size $n \times d$ containing d-dimensional datasets. Each of the d columns represents one random variable. Numbers of observations n equal the number of rows in the datasets.

Preprocessing: a character vector giving the preprocessing types. One of "histogram", "kernel density estimation" or "k-nearest neighbour".

cmax: maximum number of components $c_{\max} > 0$. The default value is 15.

cmin: minimum number of components $c_{\min} > 0$. The default value is 1.

Criterion: a character giving the information criterion type. One of default Akaike "AIC", "AIC3", "AIC4" or "AICc", Bayesian "BIC", consistent Akaike "CAIC", Hannan-Quinn "HQC", minimum description length "MDL2" or "MDL5", approximate weight of evidence "AWE", classification likelihood "CLC", integrated classification likelihood "ICL" or "ICL-BIC", partition coefficient "PC", total of positive relative deviations "D" or sum of squares error "SSE".

Variables: a character vector of length d containing types of variables. One of "continuous" or "discrete".

pdf: a character vector of length d containing continuous or discrete parametric family types. One of "normal", "lognormal", "Weibull", "gamma", "Gumbel", "binomial", "Poisson", "Dirac" or "vonMises".

theta1: a vector of length d containing initial component parameters. One of n_{il} = number of categories-1 for "binomial" distribution or "NA" otherwise.

theta2: a vector of length d containing initial component parameters. Currently not used.

K: a character or a vector or a list of vectors containing numbers of bins v for the histogram and the kernel density estimation or numbers of nearest neighbours k for the k-nearest neighbour. There is no genuine rule to identify v or k. Consequently, the REBMIX algorithm identifies them from the set K of input values by minimizing the information criterion. The Sturges rule $v = 1 + \log_2(n)$, Log_{10} rule $v = 10\log_{10}(n)$ or RootN rule $v = 2\sqrt{n}$ can be applied to estimate the limiting numbers of bins or the rule of thumb $k = \sqrt{n}$ to guess the intermediate number of nearest neighbours. If, e.g., K = c(10,20,40,60) and minimum IC coincides, e.g., 40, brackets are set to 20 and 60 and the golden section is applied to refine the minimum search. See also kseq for sequence of bins or nearest neighbours generation. The default value is "auto".

y0: a vector of length d containing origins. The default value is numeric().

ymin: a vector of length d containing minimum observations. The default value is numeric().

ymax: a vector of length d containing maximum observations. The default value is numeric().

ar: acceleration rate $0 < a_{\mathrm{r}} \leq 1$. The default value is 0.1 and in most cases does not have to be altered.

Restraints: a character giving the restraints type. One of "rigid" or default "loose". The rigid restraints are obsolete and applicable for well separated components only.

w: a list of vectors of length c containing component weights w_l summing to 1.

`Theta`: a list of lists each containing c parametric family types `pdfl`. One of `"normal"`, `"lognormal"`, `"Weibull"`, `"gamma"`, `"Gumbel"`, `"binomial"`, `"Poisson"`, `"Dirac"` or circular `"vonMises"` defined for $0 \leq y_i \leq 2\pi$. Component parameters `theta1.l` follow the parametric family types. One of μ_{il} for normal, lognormal, Gumbel and von Mises distributions and θ_{il} for Weibull, gamma, binomial, Poisson and Dirac distributions. Component parameters `theta2.l` follow `theta1.l`. One of σ_{il} for normal and lognormal distributions, β_{il} for Weibull, gamma and Gumbel distributions, p_{il} for binomial distribution and κ_{il} for von Mises distribution.

`summary`: a data frame with additional information about dataset, preprocessing, $c_{\max}$, $c_{\min}$, information criterion type, a_{r}, restraints type, optimal c, optimal v or k, K, y_{i0}, $y_{i\min}$, $y_{i\max}$, optimal h_i, information criterion IC, log likelihood $\log L$ and degrees of freedom M.

`summary.EM`: a data frame with additional information about dataset, strategy for the EM algorithm `strategy`, variant of the EM algorithm `variant`, acceleration type `acceleration`, tolerance `tolerance`, acceleration multilplier `acceleration.multiplier`, maximum allowed number of iterations `maximum.iterations`, number of iterations used for obtaining optimal solution `opt.iterations.nbr` and total number of iterations of the EM algorithm `total.iterations.nbr`.

`pos`: position in the `summary` data frame at which log likelihood $\log L$ attains its maximum.

`opt.c`: a list of vectors containing numbers of components for optimal v for the histogram and the kernel density estimation or for optimal number of nearest neighbours k for the k-nearest neighbour.

`opt.IC`: a list of vectors containing information criteria for optimal v for the histogram and the kernel density estimation or for optimal number of nearest neighbours k for the k-nearest neighbour.

`opt.logL`: a list of vectors containing log likelihoods for optimal v for the histogram and the kernel density estimation or for optimal number of nearest neighbours k for the k-nearest neighbour.

`opt.D`: a list of vectors containing totals of positive relative deviations for optimal v for the histogram and the kernel density estimation or for optimal number of nearest neighbours k for the k-nearest neighbour.

`all.K`: a list of vectors containing all processed numbers of bins v for the histogram and the kernel density estimation or all processed numbers of nearest neighbours k for the k-nearest neighbour.

`all.IC`: a list of vectors containing information criteria for all processed numbers of bins v for the histogram and the kernel density estimation or for all processed numbers of nearest neighbours k for the k-nearest neighbour.

Author(s)

Marko Nagode

`REBMIX-methods` *REBMIX Algorithm for Univariate or Multivariate Finite Mixture Estimation*

Description

Returns as default the REBMIX algorithm output for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities. If `model` equals `"REBMVNORM"` output for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices is returned.

Usage

```
## S4 method for signature 'REBMIX'
REBMIX(model = "REBMIX", Dataset = list(), Preprocessing = character(),
       cmax = 15, cmin = 1, Criterion = "AIC", pdf = character(),
       theta1 = numeric(),theta2 = numeric(), K = "auto", y0 = numeric(),
       ymin = numeric(), ymax = numeric(), ar = 0.1, Restraints = "loose",
       EMcontrol = NULL, ...)
## ... and for other signatures
## S4 method for signature 'REBMIX'
summary(object, ...)
## ... and for other signatures
```

Arguments

`model`	see Methods section below.
`Dataset`	a list of data frames of size $n \times d$ containing d-dimensional datasets. Each of the d columns represents one random variable. Numbers of observations n equal the number of rows in the datasets.
`Preprocessing`	a character vector giving the preprocessing types. One of `"histogram"`, `"kernel density estimation"` or `"k-nearest neighbour"`.
`cmax`	maximum number of components $c_{\max} > 0$. The default value is `15`.
`cmin`	minimum number of components $c_{\min} > 0$. The default value is `1`.
`Criterion`	a character giving the information criterion type. One of default Akaike `"AIC"`, `"AIC3"`, `"AIC4"` or `"AICc"`, Bayesian `"BIC"`, consistent Akaike `"CAIC"`, Hannan-Quinn `"HQC"`, minimum description length `"MDL2"` or `"MDL5"`, approximate weight of evidence `"AWE"`, classification likelihood `"CLC"`, integrated classification likelihood `"ICL"` or `"ICL-BIC"`, partition coefficient `"PC"`, total of positive relative deviations `"D"` or sum of squares error `"SSE"`.
`pdf`	a character vector of length d containing continuous or discrete parametric family types. One of `"normal"`, `"lognormal"`, `"Weibull"`, `"gamma"`, `"Gumbel"`, `"binomial"`, `"Poisson"`, `"Dirac"` or `"vonMises"`.
`theta1`	a vector of length d containing initial component parameters. One of $n_{il} =$ number of categories $- 1$ for `"binomial"` distribution or `"NA"` otherwise.
`theta2`	a vector of length d containing initial component parameters. Currently not used.
`K`	a character or a vector or a list of vectors containing numbers of bins v for the histogram and the kernel density estimation or numbers of nearest neighbours k for the k-nearest neighbour. There is no genuine rule to identify v or k. Consequently, the REBMIX algorithm identifies them from the set `K` of input values by

	minimizing the information criterion. The Sturges rule $v = 1 + \log_2(n)$, Log_{10} rule $v = 10\log_{10}(n)$ or RootN rule $v = 2\sqrt{n}$ can be applied to estimate the limiting numbers of bins or the rule of thumb $k = \sqrt{n}$ to guess the intermediate number of nearest neighbours. If, e.g., `K = c(10,20,40,60)` and minimum `IC` coincides, e.g., `40`, brackets are set to `20` and `60` and the golden section is applied to refine the minimum search. See also `kseq` for sequence of bins or nearest neighbours generation. The default value is `"auto"`.
`y0`	a vector of length d containing origins. The default value is `numeric()`.
`ymin`	a vector of length d containing minimum observations. The default value is `numeric()`.
`ymax`	a vector of length d containing maximum observations. The default value is `numeric()`.
`ar`	acceleration rate $0 < a_\mathrm{r} \leq 1$. The default value is `0.1` and in most cases does not have to be altered.
`Restraints`	a character giving the restraints type. One of `"rigid"` or default `"loose"`. The rigid restraints are obsolete and applicable for well separated components only.
`EMcontrol`	an object of class `EM.Control`.
`object`	see Methods section below.
`...`	currently not used; additional arguments affecting the summary produced.

Value

Returns an object of class `REBMIX` or `REBMVNORM`.

Methods

`signature(model = "REBMIX")` a character giving the default class name `"REBMIX"` for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities.

`signature(model = "REBMVNORM")` a character giving the class name `"REBMVNORM"` for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices.

`signature(object = "REBMIX")` an object of class `REBMIX`.

`signature(object = "REBMVNORM")` an object of class `REBMVNORM`.

Author(s)

Marko Nagode

References

H. A. Sturges. The choice of a class interval. Journal of American Statistical Association, 21(153): 65-66, 1926. https://www.jstor.org/stable/2965501.

P. F. Velleman. Interactive computing for exploratory data analysis I: display algorithms. Proceedings of the Statistical Computing Section, American Statistical Association, 1976.

W. J. Dixon and R. A. Kronmal. The Choice of origin and scale for graphs. Journal of the ACM, 12(2): 259-261, 1965. https://doi.org/10.1145/321264.321277.

M. Nagode and M. Fajdiga. A general multi-modal probability density function suitable for the rainflow ranges of stationary random processes. International Journal of Fatigue, 20(3):211-223, 1998. https://doi.org/10.1016/S0142-1123(97)00106-0.

M. Nagode and M. Fajdiga. An improved algorithm for parameter estimation suitable for mixed weibull distributions. International Journal of Fatigue, 22(1):75-80, 2000. https://doi.org/10.1016/S0142-1123(99)00112-7.

M. Nagode, J. Klemenc, and M. Fajdiga. Parametric modelling and scatter prediction of rainflow matrices. International Journal of Fatigue, 23(6):525-532, 2001. https://doi.org/10.1016/S0142-1123(01)00007-X.

M. Nagode and M. Fajdiga. An alternative perspective on the mixture estimation problem. Reliability Engineering & System Safety, 91(4):388-397, 2006. https://doi.org/10.1016/j.ress.2005.02.005.

M. Nagode and M. Fajdiga. The rebmix algorithm for the univariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(5):876-892, 2011a. https://doi.org/10.1080/03610920903480890.

M. Nagode and M. Fajdiga. The rebmix algorithm for the multivariate finite mixture estimation. Communications in Statistics - Theory and Methods, 40(11):2022-2034, 2011b. https://doi.org/10.1080/03610921003725788.

M. Nagode. Finite mixture modeling via REBMIX. Journal of Algorithms and Optimization, 3(2):14-28, 2015. https://doi.org/10.5963/JAO0302001.

Examples

```
# Generate and plot univariate normal dataset.

n <- c(998, 263, 1086, 487)

Theta <- new("RNGMIX.Theta", c = 4, pdf = "normal")

a.theta1(Theta) <- c(688, 265, 30, 934)
a.theta2(Theta) <- c(72, 54, 34, 28)

normal <- RNGMIX(Dataset.name = "complex1",
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))

normal

a.Dataset(normal, 1)[1:20,]
```

```
# Estimate number of components, component weights and component parameters.

normalest <- REBMIX(Dataset = a.Dataset(normal),
  Preprocessing = "h",
  cmax = 8,
  Criterion = "BIC",
  pdf = "n")

normalest

BIC(normalest)

logL(normalest)

# Plot finite mixture.

plot(normalest, nrow = 2, what = c("den", "dis"), npts = 1000)

# EM algorithm utilization

# Load iris data.

data(iris)

Dataset <- list(data.frame(iris[, c(1:4)]))

# Create EM.Control object.

EM <- new("EM.Control",
  strategy = "exhaustive",
  variant = "EM",
  acceleration = "fixed",
  tolerance = 1e-4,
  acceleration.multiplier = 1.0,
  maximum.iterations = 1000)

# Mixture parameter estimation using REBMIX and EM algorithm.

irisest <- REBMIX(model="REBMVNORM",
  Dataset = Dataset,
  Preprocessing = "histogram",
  cmax = 10,
  Criterion = "BIC",
  EMcontrol = EM)

irisest

# Print total number of EM iterations used in Ehxaustive strategy from summary.EM slot.

a.summary.EM(irisest, pos = 1, col.name = "total.iterations.nbr")
```

REBMIX.boot-class *Class* "REBMIX.boot"

Description

Object of class REBMIX.boot.

Objects from the Class

Objects can be created by calls of the form new("REBMIX.boot",...). Accessor methods for the slots are a.rseed(x = NULL), a.pos(x = NULL), a.Bootstrap(x = NULL), a.B(x = NULL), a.n(x = NULL), a.replace(x = NULL), a.prob(x = NULL), a.c(x = NULL), a.c.se(x = NULL), a.c.cv(x = NULL), a.c.mode(x = NULL), a.c.prob(x = NULL), a.w(x = NULL), a.w.se(x = NULL), a.w.cv(x = NULL), a.Theta(x = NULL), a.Theta.se(x = NULL) and a.Theta.cv(x = NULL), where x stands for an object of class REBMIX.boot.

Slots

x: an object of class REBMIX.

rseed: set the random seed to any negative integer value to initialize the sequence. The first bootstrap dataset corresponds to it. For each next bootstrap dataset the random seed is decremented $r_{\text{seed}} = r_{\text{seed}} - 1$. The default value is -1.

pos: a desired row number in x@summary to be bootstrapped. The default value is 1.

Bootstrap: a character giving the bootstrap type. One of default "parametric" or "nonparametric".

B: number of bootstrap datasets. The default value is 100.

n: number of observations. The default value is numeric().

replace: logical. The sampling is with replacement if TRUE, see also sample. The default value is TRUE.

prob: a vector of length n containing probability weights, see also sample. The default value is numeric().

c: a vector containing numbers of components for B bootstrap datasets.

c.se: standard error of numbers of components c.

c.cv: coefficient of variation of numbers of components c.

c.mode: mode of numbers of components c.

c.prob: probability of mode c.mode.

w: a matrix containing component weights for $\leq B$ bootstrap datasets.

w.se: a vector containing standard errors of component weights w.

w.cv: a vector containing coefficients of variation of component weights w.

Theta: a list of matrices containing component parameters theta1.1 and theta2.1 for $\leq B$ bootstrap datasets.

Theta.se: a list of vectors containing standard errors of component parameters theta1.1 and theta2.1.

Theta.cv: a list of vectors containing coefficients of variation of component parameters theta1.1 and theta2.1.

Author(s)

Marko Nagode

RNGMIX-class *Class* "RNGMIX"

Description

Object of class RNGMIX.

Objects from the Class

Objects can be created by calls of the form new("RNGMIX", ...). Accessor methods for the slots are a.Dataset.name(x = NULL), a.rseed(x = NULL), a.n(x = NULL), a.Theta(x = NULL), a.Dataset(x = NULL,pos = 0), a.Zt(x = NULL), a.w(x = NULL), a.Variables(x = NULL), a.ymin(x = NULL) and a.ymax(x = NULL), where x and pos stand for an object of class RNGMIX and a desired slot item, respectively.

Slots

Dataset.name: a character vector containing list names of data frames of size $n \times d$ that d-dimensional datasets are written in.

rseed: set the random seed to any negative integer value to initialize the sequence. The first file in Dataset.name corresponds to it. For each next file the random seed is decremented $r_{\text{seed}} = r_{\text{seed}} - 1$. The default value is -1.

n: a vector containing numbers of observations in classes n_l, where number of observations $n = \sum_{l=1}^{c} n_l$.

Theta: a list containing c parametric family types pdfl. One of "normal", "lognormal", "Weibull", "gamma", "Gumbel", "binomial", "Poisson", "Dirac" or circular "vonMises" defined for $0 \leq y_i \leq 2\pi$. Component parameters theta1.l follow the parametric family types. One of μ_{il} for normal, lognormal, Gumbel and von Mises distributions and θ_{il} for Weibull, gamma, binomial, Poisson and Dirac distributions. Component parameters theta2.l follow theta1.l. One of σ_{il} for normal and lognormal distributions, β_{il} for Weibull, gamma and Gumbel distributions, p_{il} for binomial distribution, κ_{il} for von Mises distribution and "NA" otherwise.

Dataset: a list of data frames of size $n \times d$ containing d-dimensional datasets. Each of the d columns represents one random variable. Numbers of observations n equal the number of rows in the datasets.

Zt: a factor of true cluster membership.

w: a vector of length c containing component weights w_l summing to 1.

Variables: a character vector containing types of variables. One of "continuous" or "discrete".

ymin: a vector of length d containing minimum observations.

ymax: a vector of length d containing maximum observations.

Author(s)

Marko Nagode

RNGMIX-methods *Random Univariate or Multivariate Finite Mixture Generation*

Description

Returns as default the RNGMIX univariate or multivariate random datasets for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac, uniform or von Mises component densities. If `model` equals `"RNGMVNORM"` multivariate random datasets for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices are returned.

Usage

```
## S4 method for signature 'RNGMIX'
RNGMIX(model = "RNGMIX", Dataset.name = character(),
       rseed = -1, n = numeric(), Theta = list(), ...)
## ... and for other signatures
```

Arguments

`model`	see Methods section below.
`Dataset.name`	a character vector containing list names of data frames of size $n \times d$ that d-dimensional datasets are written in.
`rseed`	set the random seed to any negative integer value to initialize the sequence. The first file in `Dataset.name` corresponds to it. For each next file the random seed is decremented $r_{\text{seed}} = r_{\text{seed}} - 1$. The default value is `-1`.
`n`	a vector containing numbers of observations in classes n_l, where number of observations $n = \sum_{l=1}^{c} n_l$.
`Theta`	a list containing c parametric family types `pdfl`. One of `"normal"`, `"lognormal"`, `"Weibull"`, `"gamma"`, `"Gumbel"`, `"binomial"`, `"Poisson"`, `"Dirac"`, `"uniform"` or circular `"vonMises"` defined for $0 \leq y_i \leq 2\pi$. Component parameters `theta1.l` follow the parametric family types. One of μ_{il} for normal, lognormal, Gumbel and von Mises distributions and θ_{il} for Weibull, gamma, binomial, Poisson and Dirac distributions. Component parameters `theta2.l` follow `theta1.l`. One of σ_{il} for normal and lognormal distributions, β_{il} for Weibull, gamma and Gumbel distributions, p_{il} for binomial distribution, κ_{il} for von Mises distribution and `"NA"` otherwise.
`...`	currently not used.

Details

RNGMIX is based on the "Minimal" random number generator `ran1` of Park and Miller with the Bays-Durham shuffle and added safeguards that returns a uniform random deviate between 0.0 and 1.0 (exclusive of the endpoint values).

Value

Returns an object of class RNGMIX or RNGMVNORM.

Methods

signature(model = "RNGMIX") a character giving the default class name "RNGMIX" for mixtures of conditionally independent normal, lognormal, Weibull, gamma, Gumbel, binomial, Poisson, Dirac or von Mises component densities.

signature(model = "RNGMVNORM") a character giving the class name "RNGMVNORM" for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices.

Author(s)

Marko Nagode

References

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press, Cambridge, 1992.

Examples

```
devAskNewPage(ask = TRUE)

# Generate and print multivariate normal datasets with diagonal
# variance-covariance matrices.

n <- c(75, 100, 125, 150, 175)

Theta <- new("RNGMIX.Theta", c = 5, pdf = rep("normal", 4))

a.theta1(Theta, 1) <- c(10, 12, 10, 12)
a.theta1(Theta, 2) <- c(8.5, 10.5, 8.5, 10.5)
a.theta1(Theta, 3) <- c(12, 14, 12, 14)
a.theta1(Theta, 4) <- c(13, 15, 7, 9)
a.theta1(Theta, 5) <- c(7, 9, 13, 15)
a.theta2(Theta, 1) <- c(1, 1, 1, 1)
a.theta2(Theta, 2) <- c(1, 1, 1, 1)
a.theta2(Theta, 3) <- c(1, 1, 1, 1)
a.theta2(Theta, 4) <- c(2, 2, 2, 2)
a.theta2(Theta, 5) <- c(3, 3, 3, 3)

simulated <- RNGMIX(Dataset.name = paste("simulated_", 1:25, sep = ""),
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))

simulated

plot(simulated, pos = 22, nrow = 2, ncol = 3)
```

```
# Generate and print multivariate normal datasets with unrestricted
# variance-covariance matrices.

n <- c(200, 50, 50)

Theta <- new("RNGMVNORM.Theta", c = 3, d = 3)

a.theta1(Theta, 1) <- c(0, 0, 0)
a.theta1(Theta, 2) <- c(-6, 3, 6)
a.theta1(Theta, 3) <- c(6, 6, 4)
a.theta2(Theta, 1) <- c(9, 0, 0, 0, 4, 0, 0, 0, 1)
a.theta2(Theta, 2) <- c(4, -3.2, -0.2, -3.2, 4, 0, -0.2, 0, 1)
a.theta2(Theta, 3) <- c(4, 3.2, 2.8, 3.2, 4, 2.4, 2.8, 2.4, 2)

simulated <- RNGMIX(model = "RNGMVNORM",
  Dataset.name = paste("simulated_", 1:2, sep = ""),
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))

simulated

plot(simulated, pos = 2, nrow = 3, ncol = 1)

# Generate and print multivariate mixed continuous-discrete datasets.

n <- c(400, 100, 500)

Theta <- new("RNGMIX.Theta", c = 3, pdf = c("lognormal", "Poisson", "binomial", "Weibull"))

a.theta1(Theta, 1) <- c(1, 2, 10, 2)
a.theta1(Theta, 2) <- c(3.5, 10, 10, 10)
a.theta1(Theta, 3) <- c(2.5, 15, 10, 25)
a.theta2(Theta, 1) <- c(0.3, NA, 0.9, 3)
a.theta2(Theta, 2) <- c(0.2, NA, 0.1, 7)
a.theta2(Theta, 3) <- c(0.4, NA, 0.7, 20)

simulated <- RNGMIX(Dataset.name = paste("simulated_", 1:5, sep = ""),
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))

simulated

plot(simulated, pos = 4, nrow = 2, ncol = 3)

# Generate and print univariate mixed Weibull dataset.

n <- c(75, 100, 125, 150, 175)

Theta <- new("RNGMIX.Theta", c = 5, pdf = "Weibull")

a.theta1(Theta) <- c(12, 10, 14, 15, 9)
```

```
a.theta2(Theta) <- c(2, 4.1, 3.2, 7.1, 5.3)

simulated <- RNGMIX(Dataset.name = "simulated",
  rseed = -1,
  n = n,
  Theta = a.Theta(Theta))

simulated

plot(simulated, pos = 1)

# Generate and print multivariate normal datasets with unrestricted
# variance-covariance matrices.

# Set dimension, dataset size, number of components and seed.

d <- 2; n <- 1000; c <- 10; set.seed(123)

# Component weights are generated.

w <- runif(c, 0.1, 0.9); w <- w / sum(w)

# Set range of means and rang of eigenvalues.

mu <- c(-100, 100); lambda <- c(1, 100)

# Component means and variance-covariance matrices are calculated.

Mu <- list(); Sigma <- list()
for (l in 1:c) {
  Mu[[l]] <- runif(d, mu[1], mu[2])
  Lambda <- diag(runif(d, lambda[1], lambda[2]), nrow = d, ncol = d)
  P <- svd(matrix(runif(d * d, -1, 1), nc = d))$u
  Sigma[[l]] <- P
}

# Numbers of observations are calculated and component means and
# variance-covariance matrices are stored.

n <- round(w * n); Theta <- list()
for (l in 1:c) {
  Theta[[paste0("pdf", l)]] <- rep("normal", d)
  Theta[[paste0("theta1.", l)]] <- Mu[[l]]
  Theta[[paste0("theta2.", l)]] <- as.vector(Sigma[[l]])
}

# Dataset is generated.

simulated <- RNGMIX(model = "RNGMVNORM", Dataset.name = "mvnorm_1",
  rseed = -1, n = n, Theta = Theta)

plot(simulated)
```

RNGMIX.Theta-class *Class* "RNGMIX.Theta"

Description

Object of class RNGMIX.Theta.

Objects from the Class

Objects can be created by calls of the form new("RNGMIX.Theta",...). Accessor methods for the slots are a.c(x = NULL), a.d(x = NULL), a.pdf(x = NULL) and a.Theta(x = NULL), where x stands for an object of class RNGMIX.Theta. Setter methods a.theta1(x = NULL,l = numeric()) and a.theta2(x = NULL,l = numeric()) are provided to write to Theta slot, where $l = 1, \ldots, c$.

Slots

c: number of components $c > 0$. The default value is 1.

d: number of dimensions.

pdf: a character vector of length d containing continuous or discrete parametric family types. One of "normal", "lognormal", "Weibull", "gamma", "Gumbel", "binomial", "Poisson", "Dirac" or "vonMises".

Theta: a list containing c parametric family types pdf1. One of "normal", "lognormal", "Weibull", "gamma", "Gumbel", "binomial", "Poisson", "Dirac" or circular "vonMises" defined for $0 \leq y_i \leq 2\pi$. Component parameters theta1.l follow the parametric family types. One of μ_{il} for normal, lognormal, Gumbel and von Mises distributions and θ_{il} for Weibull, gamma, binomial, Poisson and Dirac distributions. Component parameters theta2.l follow theta1.l. One of σ_{il} for normal and lognormal distributions, β_{il} for Weibull, gamma and Gumbel distributions, p_{il} for binomial distribution, κ_{il} for von Mises distribution and "NA" otherwise.

Author(s)

Marko Nagode

Examples

```
Theta <- new("RNGMIX.Theta", c = 2, pdf = c("normal", "Weibull"))

a.theta1(Theta, l = 1) <- c(2, 10)
a.theta2(Theta, l = 1) <- c(0.5, 2.3)
a.theta1(Theta, l = 2) <- c(20, 50)
a.theta2(Theta, l = 2) <- c(3, 4.2)

Theta

Theta <- new("RNGMVNORM.Theta", c = 2, d = 3)

a.theta1(Theta, l = 1) <- c(2, 10, -20)
a.theta1(Theta, l = 2) <- c(-2.4, -15.1, 30)
```

Theta

---

split-methods *Splits Dataset into Train and Test Datasets*

Description

Returns (invisibly) the object containing train and test observations $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ as well as true class membership $\boldsymbol{\Omega}_g$ for the test dataset.

Usage

```
## S4 method for signature 'numeric'
split(p = 0.75, Dataset = data.frame(), class = numeric(), ...)
## S4 method for signature 'list'
split(p = list(), Dataset = data.frame(), class = numeric(), ...)
## ... and for other signatures
```

Arguments

p	see Methods section below.
Dataset	a data frame containing dataset Y of length n. For the dataset the corresponding class membership $\boldsymbol{\Omega}_g$ is known. The default value is data.frame().
class	a column number in Dataset containing the class membership information. The default value is numeric().
...	further arguments to sample.

Value

Returns an object of class RCLS.chunk.

Methods

signature(p = "numeric") a number specifying the fraction of observations for training $0.0 \leq p \leq 1.0$. The default value is 0.75.

signature(p = "list") a list composed of column number p$type in Dataset containing the type membership information followed by the corresponding train p$train and test p$test values. The default value is list().

Author(s)

Marko Nagode

Examples

```
data("iris")

# Split dataset into train (75%) and test (25%) subsets.

set.seed(5)

Iris <- split(p = 0.75, Dataset = iris, class = 5)

Iris

# Generate simulated dataset.

N <- 1000

class <- c(rep("A", 0.4 * N), rep("B", 0.2 * N),
  rep("C", 0.1 * N), rep("D", 0.05 * N), rep("E", 0.25 * N))

type <- c(rep("train", 0.75 * N), rep("test", 0.25 * N))

n <- 30

Dataset <- data.frame(1:n, sample(class, n))

colnames(Dataset) <- c("y", "class")

# Split dataset into train (60%) and test (40%) subsets.

simulated <- split(p = 0.6, Dataset = Dataset, class = 2)

simulated

# Generate simulated dataset.

Dataset <- data.frame(1:n, sample(class, n), sample(type, n))

colnames(Dataset) <- c("y", "class", "type")

# Split dataset into train and test subsets.

simulated <- split(p = list(type = 3, train = "train",
  test = "test"), Dataset = Dataset, class = 2)

simulated
```

`SSE-methods` *Sum of Squares Error*

Description

Returns the sum of squares error at pos.

Usage

```
## S4 method for signature 'REBMIX'
SSE(x = NULL, pos = 1, ...)
## ... and for other signatures
```

Arguments

x	see Methods section below.
pos	a desired row number in x@summary for which the information criterion is calculated. The default value is 1.
...	currently not used.

Methods

signature(x = "REBMIX") an object of class REBMIX.

signature(x = "REBMVNORM") an object of class REBMVNORM.

Author(s)

Marko Nagode

References

C. M. Bishop. Neural Networks for Pattern Recognition. Clarendon Press, Oxford, 1995.

---

truck *Truck Dataset*

Description

The dataset contains amplitudes and means measured on a truck wheels.

Usage

```
data("truck")
```

Format

truck is a data frame with 31665 rows and 2 variables (columns) named:

1. Amplitude continuous.
2. Mean continuous.

Author(s)

Mitja Franko

Examples

```
data("truck")
```

`weibull` *Weibull Dataset 8.1*

Description

The complete data are the failure times in weeks.

Usage

```
data("weibull")
```

Format

`weibull` is a data frame with 50 cases (rows) and 1 variables (columns) named:

1. `Failure.Time` continuous.

References

D. N. P. Murthy, M. Xie, and R. Jiang. Weibull Models. John Wiley & Sons, New York, 2003.

Examples

```
data("weibull")
```

`weibullnormal` *Weibull-normal Simulated Dataset*

Description

The dataset contains amplitudes and means simulated from a three component Weibull-normal mixture.

Usage

```
data("weibullnormal")
```

Format

`weibullnormal` is a data frame with 10000 rows and 2 variables (columns) named:

1. `Amplitude` continuous.
2. `Mean` continuous.

Author(s)

Mitja Franko

Examples

```
data("weibullnormal")
```

`wine` *Wine Recognition Data*

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 (1-3). The analysis determined the quantities of 13 constituents: alcohol, malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, colour intensity, hue, OD280/OD315 of diluted wines, and proline found in each of the three types of the wines. The number of instances in classes 1 to 3 is 59, 71 and 48, respectively.

Usage

```
data("wine")
```

Format

`wine` is a data frame with 178 cases (rows) and 14 variables (columns) named:

1. `Alcohol` continuous.
2. `Malic.Acid` continuous.
3. `Ash` continuous.
4. `Alcalinity.of.Ash` continuous.
5. `Magnesium` continuous.
6. `Total.Phenols` continuous.
7. `Flavanoids` continuous.
8. `Nonflavanoid.Phenols` continuous.
9. `Proanthocyanins` continuous.
10. `Color.Intensity` continuous.
11. `Hue` continuous.
12. `OD280.OD315.of.Diluted.Wines` continuous.
13. `Proline` continuous.
14. `Cultivar` discrete `1`, `2` or `3`.

Source

A. Asuncion and D. J. Newman. Uci machine learning repository, 2007. http://archive.ics.uci.edu/ml.

References

S. J. Roberts, R. Everson, and I. Rezek. Maximum certainty data partitioning. Pattern Recognition, 33(5):833-839, 2000. https://doi.org/10.1016/S0031-3203(99)00086-2.

Examples

```
## Not run:
devAskNewPage(ask = TRUE)

data("wine")

# Show level attributes.

levels(factor(wine[["Cultivar"]]))

# Split dataset into train (75

set.seed(3)

Wine <- split(p = 0.75, Dataset = wine, class = 14)

# Estimate number of components, component weights and component
# parameters for train subsets.

n <- range(a.ntrain(Wine))

K <- c(as.integer(1 + log2(n[1])), # Minimum v follows Sturges rule.
  as.integer(10 * log10(n[2]))) # Maximum v follows log10 rule.

K <- c(floor(K[1]^(1/13)), ceiling(K[2]^(1/13)))

wineest <- REBMIX(model = "REBMVNORM",
  Dataset = a.train(Wine),
  Preprocessing = "kernel density estimation",
  cmax = 10,
  Criterion = "ICL-BIC",
  pdf = rep("normal", 13),
  K = K[1]:K[2],
  Restraints = "loose")

plot(wineest, pos = 1, nrow = 7, ncol = 6, what = c("den"))
plot(wineest, pos = 2, nrow = 7, ncol = 6, what = c("den"))
plot(wineest, pos = 3, nrow = 7, ncol = 6, what = c("den"))

# Selected chunks.

winecla <- RCLSMIX(model = "RCLSMVNORM",
```

```
  x = list(wineest),
  Dataset = a.test(Wine),
  Zt = a.Zt(Wine))

winecla

summary(winecla)

# Plot selected chunks.

plot(winecla, nrow = 7, ncol = 6)

## End(Not run)
```

Index