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 ................................................................. 2
  AIC-methods .......................................................... 4
  AWE-methods .......................................................... 5
  BFSMIX-methods ...................................................... 6
  BIC-methods ......................................................... 7
  boot-methods .......................................................... 8
  chunk-methods ......................................................... 10
  CLC-methods .......................................................... 11
  demix-methods ......................................................... 11
  dfmix-methods ........................................................ 13
  EM.Control-class ...................................................... 15
  galaxy ............................................................... 17
  HQC-methods ........................................................... 18
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")
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.
8. Occupation one of the 14 discrete values tech-support, craft-repair, other-service, sales, exec-managerial, prof-specialty, handlers-cleaners, machine-op-inspect, 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.
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, colombia, hungary, guatemala, nicaragua, scotland, thailand, yugoslavia, el-salvador, trinidad&tobago, peru, hong or holand-netherlands.
16. Income binary <=50k or >50k.

Source


References

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**

```r
## 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


---

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.
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

```r
## 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 \( Y_{train} \) all of length \( n_{train} \). For the train datasets the corresponding class membership \( \Omega_g \) is known. This yields \( n_{train} = \sum_{g=1}^{s} n_{train} \), while \( Y_{train} \cap Y_{train} = \emptyset \) for all \( g \neq g \). 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 \( \Omega_g \) is not known. The default value is data.frame().
- `Zt` a factor of true class membership \( \Omega_g \) 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


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.

Methods

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

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

Author(s)

Marko Nagode
References


---

Boot-Method

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 RNGMVNM the boot output for mixtures of multivariate normal component densities with unrestricted variance-covariance matrices is returned.

Usage

```r
## 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


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)
Extracts Chunk from Train and Test Datasets

Description
Returns (invisibly) the object containing train and test observations \( x_1, \ldots, x_n \) as well as true class membership \( \Omega_g \) for the test dataset. Vectors \( x \) are subvectors of \( y = (y_1, \ldots, y_d)^\top \).

Usage
```r
## 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 \( x \). The default value is \( 1:d \).

Value
Returns an object of class RCLS.chunk.

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

Author(s)
Marko Nagode

Examples
```r
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
```
Description

Returns the classification likelihood criterion at pos.

Usage

```r
## 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


demix-methods  |  Empirical Density Calculation

Description

Returns the data frame containing observations $x_1, \ldots, x_n$ and empirical densities $f_1, \ldots, f_n$ for the kernel density estimation or $k$-nearest neighbour or bin means $\bar{x}_1, \ldots, \bar{x}_v$ and empirical densities $f_1, \ldots, f_v$ for the histogram preprocessing. Vectors $x$ and $\hat{x}$ are subvectors of $y = (y_1, \ldots, y_d)^T$ and $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_d)^T$. 
Usage

```r
## 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 `x` or `\bar{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


Examples

```r
# 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 \(x_1, \ldots, x_n\) and predictive marginal densities \(f(x|c, w, \Theta)\). Vectors \(x\) are subvectors of \(y = (y_1, \ldots, y_d)^T\).

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 \( y = (y_1, \ldots, y_d)^T \) 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 \( x \). 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


Examples

# Generate simulated dataset.

n <- c(15, 15)
Theta <- new("RNGMIX.Theta", c = 2, pdf = rep("normal", 3))

a.thetal(Theta, 1) <- c(10, 20, 30)
a.thetal(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.
EM.Control-class

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" or "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 $a_{EM}$, $1.0 \leq a_{EM} \leq 2.0$. 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


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
**galaxy**

---

**Description**

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

**Usage**

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

**Format**

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

**Source**


**References**


HQC-methods  

**Hannan-Quinn Information Criterion**

**Description**

Returns the Hannan-Quinn information criterion at pos.

**Usage**

```r
## 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**


---

ICL-methods  

**Integrated Classification Likelihood Criterion**

**Description**

Returns the integrated classification likelihood criterion at pos.
Usage

```r
## 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


---

**ICLBIC-methods**

*Approximate Integrated Classification Likelihood Criterion*

Description

Returns the approximate integrated classification likelihood criterion at pos.

Usage

```r
## 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


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:

2. Sepal.Width continuous.
5. Class discrete iris-setosa, iris-versicolour or iris-virginica.

Source


References

Examples

```r
## 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

```r
# 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

```r
## 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

Methods

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

Author(s)

Marko Nagode

References


Empirical Distribution Function Calculation

Description

Returns the data frame containing observations \( x_1, \ldots, x_n \) and empirical distribution functions \( F_1, \ldots, F_n \). Vectors \( x \) are subvectors of \( y = (y_1, \ldots, y_d)^T \).

Usage

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

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 \( x \). 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


Examples

# Generate simulated dataset.

n <- c(15, 15)
Theta <- new("RNGMIX.Theta", c = 2, pdf = rep("normal", 3))
a.thet1(Theta, 1) <- c(10, 20, 30)
a.thet1(Theta, 2) <- c(3, 4, 5)
a.thet2(Theta, 1) <- c(3, 2, 1)
a.thet2(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))

# 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 $x_1, \ldots, x_n$ and predictive marginal distribution functions $F(x|c, w, \Theta)$. Vectors $x$ are subvectors of $y = (y_1, \ldots, y_d)^T$.

**Usage**

```r
## 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 $y = (y_1, \ldots, y_d)^T$ 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 $x$. 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**


Examples

```r
# 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])
```
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

```r
## 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 \( \text{par} \). The default value is 0.8.

**fg**
a colour used for things like axes and boxes around plots, see also \( \text{par} \). The default value is "black".

**lty**
a line type, see also \( \text{par} \). The default value is "solid".

**lwd**
a line width, see also \( \text{par} \). 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 \( \text{par} \). 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 \( \text{par}("\text{cex}\)\). NULL and NA are equivalent to 1.0. Note that this does not affect annotation, see also \( \text{plot.default} \). The default value is 0.8.

**plot.pch**
a vector of plotting characters or symbols, see also \( \text{points} \). The default value is 19.

**contour.drawlabels**
logical. The contours are labelled if \( \text{TRUE} \). The default value is \( \text{FALSE} \).

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

**contour.method**
a character specifying where the labels will be located. The possible values are "simple", "edge" and default "flattest", see also \( \text{contour} \).

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

... further arguments to \( \text{par} \).

**Value**

Returns (invisibly) a list containing graphical parameters \( \text{par} \). Such a list can be passed as an argument to \( \text{par} \) 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


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",
                   ...
\texttt{pdf = rep("normal", d),}  
\texttt{K = K[1]:K[2])}  

\# Plot finite mixture.

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

\# End(Not run)

---

**Description**

Returns the total of positive relative deviations \( D \) at \( pos \).

**Usage**

\# S4 method for signature 'REBMIX'

\texttt{PRD(x = NULL, pos = 1, \ldots)}  
\# \ldots and for other signatures

**Arguments**

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

**Methods**

- \texttt{signature(x = "REBMIX") an object of class REBMIX.}
- \texttt{signature(x = "REBMVNORM") an object of class REBMVNORM.}

**Author(s)**

Marko Nagode

**References**


RCLRMIX-class


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 pl 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 x Ki containing cluster conditional probabilities πilk. Let πilk denote the cluster conditional probability that an observation in cluster l = 1,...,c produces the kth outcome on the ith variable. Suppose we observe i = 1,...,d polytomous categorical variables (the manifest variables), each of which contains Ki possible outcomes for observations j = 1,...,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 independent within clusters and that y1,...,yn stands for an observed d dimensional dataset of size n of vector observations yj = (y1j, ..., yij, ..., ydj)T. 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_l(y_j) and predictive N_p(y_j) frequencies calculated for unique y_j ∈ {y_1,...,y_n}, where j = 1,...,n and n ≤ n.

tau: a matrix of size n x c containing conditional probabilities τjl that observations y_1,...,y_n arise from clusters 1,...,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


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)
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


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.
```r
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 `Ω = {Ω_g; g = 1, ..., s}`.
- `levels`: a character vector of length `s` containing class names `Ω_g`.
- `ntrain`: a vector of length `s` containing numbers of observations in train datasets `Y_train^g`.
- `train`: a list of data frames containing train datasets `Y_train^g` of length `n_train^g`.
- `Zr`: a list of factors of true class membership `Ω_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 `Ω_g` for the test dataset.

**Author(s)**

Marko Nagode

**References**

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

- **ο**: number of chunks `ο`.

- **Y**: an observed `d`-dimensional dataset of size `n` of vector observations `y_j = (y_{1j}, \ldots, y_{dj})^\top` and is partitioned into train and test datasets. Vector observations `y_j` may further be split into `ο` chunks when running REBMIX, e.g., for `d = 6` and `ο = 3` the set of chunks substituting `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_{\text{test}}`. For the test dataset the corresponding class membership `Ω_y` is not known.

- **s**: finite set of size `s` of classes `Ω = \{Ω_g; g = 1, \ldots, s\}`.

- **ntrain**: a vector of length `s` containing numbers of observations in train datasets `Y_{\text{train}}`.

- **P**: a vector of length `s` containing prior probabilities `P(Ω_g) = \frac{n_{\text{train}}^g}{n_{\text{train}}}`.

- **ntest**: number of observations in test dataset `Y_{\text{test}}`.

- **Zt**: a factor of true class membership `Ω_y` for the test dataset.

- **Zp**: a factor of predictive class membership `Ω_y` 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. `Accuracy = \frac{\sum_{q=1}^{s} x_{qg}}{n_{\text{test}}}`.

- **Error**: proportion of all test observations that are classified wrongly. `Error = 1 - Accuracy`.

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

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

Chunks: a vector containing selected chunks.

Author(s)
Marko Nagode

References
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


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 =
REBMIX-class

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".

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

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

**Criterion**: a character giving the information criterion type. One of default Akaike "AIC", "AIC3", "AIC4" or "CAIC", 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".

$\theta_1$: a vector of length $d$ containing initial component parameters. One of $n_{\text{KL}} = \text{number of categories} - 1$ for "binomial" distribution or "NA" otherwise.

$\theta_2$: 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".

$y_0$: a vector of length $d$ containing origins. The default value is numeric().

$y_{\text{min}}$: a vector of length $d$ containing minimum observations. The default value is numeric().

$y_{\text{max}}$: a vector of length $d$ containing maximum observations. The default value is numeric().

$ar$: acceleration rate $0 < a_r \leq 1$. 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 \( \text{pdf} \). One of "normal", "lognormal", "Weibull", "gamma", "Gumbel", "binomial", "Poisson", "Dirac" or circular "vonMises" defined for \( 0 \leq y_i \leq 2\pi \). Component parameters \( \theta_1 \) follow the parametric family types. One of \( \mu_i \) for normal, lognormal, Gumbel and von Mises distributions and \( \theta_i \) for Weibull, gamma, binomial, Poisson and Dirac distributions. Component parameters \( \theta_2 \) follow \( \theta_1 \). One of \( \sigma_i \) for normal and lognormal distributions, \( \beta_i \) for Weibull, gamma and Gumbel distributions, \( p_i \) for binomial distribution and \( \kappa_i \) for von Mises distribution.

summary: a data frame with additional information about dataset, preprocessing, \( c_{\text{max}}, c_{\text{min}} \), information criterion type, \( a_T \), restraints type, optimal \( c \), optimal \( v \) or \( k \), \( y_{i0}, y_{i\text{min}}, y_{i\text{max}} \), 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 multiplier 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
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

```r
## 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, ...)
```

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 \( c_{\text{max}} > 0 \). The default value is 15.
- `cmin` minimum number of components \( c_{\text{min}} > 0 \). 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 \( n_{\text{cat}} = \text{number of categories} - 1 \) 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 $v = 1 + \log_2(n)$, Log10 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 $0 \leq a_r \leq 1$. 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.

... 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


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.

```r
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 Eimhaustive 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 \( r_{seed} = r_{seed} - 1 \). 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.l and theta2.l for \( \leq B \) bootstrap datasets.

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

Theta.cv: a list of vectors containing coefficients of variation of component parameters theta1.l and theta2.l.
Author(s)
Marko Nagode

RNGMIX-class

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 aymax(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 \( r_{seed} = r_{seed} - 1 \). The default value is -1.
- n: a vector containing numbers of observations in classes \( n_l \), where number of observations \( n = \sum_{i=1}^{c} n_l \).
- 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 theta1.1 follow the parametric family types. One of \( \mu_{il} \) for normal, lognormal, Gumbel and von Mises distributions and \( \theta_{il} \) for Weibull, gamma, binomial, Poisson and Dirac distributions. Component parameters theta2.1 follow theta1.1. One of \( \sigma_{il} \) for normal and lognormal distributions, \( \beta_{il} \) for Weibull, gamma and Gumbel distributions, \( p_{il} \) for binomial distribution, \( \kappa_{il} \) 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
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

```r
## 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 $r_{seed} = r_{seed} - 1$. 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 `pdf`l. 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 $\mu_{il}$ for normal, lognormal, Gumbel and von Mises distributions and $\beta_{il}$ for Weibull, gamma, binomial, Poisson and Dirac distributions. Component parameters `theta2.l` follow `theta1.l`. One of $\sigma_{il}$ for normal and lognormal distributions, $\beta_{il}$ for Weibull, gamma and Gumbel distributions, $\mu_{il}$ for binomial distribution, $\kappa_{il}$ 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


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.theta2(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 range 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 `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 $\mu_{il}$ for normal, lognormal, Gumbel and von Mises distributions and $\theta_{il}$ for Weibull, gamma, binomial, Poisson and Dirac distributions. Component parameters `theta2.l` follow `theta1.l`. One of $\sigma_{il}$ for normal and lognormal distributions, $\beta_{il}$ for Weibull, gamma and Gumbel distributions, $p_{il}$ for binomial distribution, $\kappa_{il}$ 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("RNGMNORM.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)
Description

Returns (invisibly) the object containing train and test observations $y_1, \ldots, y_n$ as well as true class membership $\Omega_g$ for the test dataset.

Usage

```r
## 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 $\Omega_g$ 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

<table>
<thead>
<tr>
<th>SSE-methods</th>
<th>Sum of Squares Error</th>
</tr>
</thead>
</table>

Description

Returns the sum of squares error at pos.
truck

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


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

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 de-
erived 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 phen-
ols, 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.
3. Ash continuous.
5. Magnesium continuous.
7. Flavanoids continuous.
11. Hue continuous.
13. Proline continuous.
14. Cultivar discrete 1, 2 or 3.
Source


References


Examples

```r
## 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

* Topic **bootstrap**
  boot-methods, 8

* Topic **classes**
  EM.Control-class, 15
  RCLRMIX-class, 33
  RCLS.chunk-class, 37
  RCLSMIX-class, 38
  REBMIX-class, 41
  REBMIX.boot-class, 48
  RNGMIX-class, 49
  RNGMIX.Theta-class, 54

* Topic **classification**
  BFSMIX-methods, 6
  chunk-methods, 10
  RCLSMIX-methods, 39
  split-methods, 55

* Topic **clustering**
  RCLRMIX-methods, 35

* Topic **datasets**
  adult, 2
  galaxy, 17
  iris, 20
  truck, 57
  weibull, 58
  weibullnormal, 58
  wine, 59

* Topic **distributions**
  demix-methods, 11
  dfmix-methods, 13
  pemix-methods, 25
  pfmix-methods, 27

* Topic **information criterion**
  AIC-methods, 4
  AWE-methods, 5
  BIC-methods, 7
  CLC-methods, 11
  HQC-methods, 18
  ICL-methods, 18
  ICLLBIC-methods, 19
  logL, 22
  MDL-methods, 23
  PC-methods, 24
  PRD-methods, 32
  SSE-methods, 56

* Topic **parameter estimation**
  kseq, 21
  REBMIX-methods, 43

* Topic **plot**
  plot-methods, 29

* Topic **random number generation**
  RNGMIX-methods, 50
  adult, 2
  AIC (AIC-methods), 4
  AIC,REBMIX-method (AIC-methods), 4
  AIC,REBMVNORM-method (AIC-methods), 4
  AIC-methods, 4
  AIC3 (AIC-methods), 4
  AIC3,REBMIX-method (AIC-methods), 4
  AIC3,REBMVNORM-method (AIC-methods), 4
  AIC3-methods (AIC-methods), 4
  AIC4 (AIC-methods), 4
  AIC4,REBMIX-method (AIC-methods), 4
  AIC4,REBMVNORM-method (AIC-methods), 4
  AIC4-methods (AIC-methods), 4
  AICC (AIC-methods), 4
  AICC,REBMIX-method (AIC-methods), 4
  AICC,REBMVNORM-method (AIC-methods), 4
  AICC-methods (AIC-methods), 4
  AWE (AWE-methods), 5
  AWE,REBMIX-method (AWE-methods), 5
  AWE,REBMVNORM-method (AWE-methods), 5
  AWE-methods, 5

  BFSMIX (BFSMIX-methods), 6
  BFSMIX,RCLSMIX-method (BFSMIX-methods), 6
  BFSMIX,REBMIX-method (BFSMIX-methods), 6
BFMSMIX-methods, 6
BIC (BIC-methods), 7
BIC, REBMIX-method (BIC-methods), 7
BIC, REBMVNORM-method (BIC-methods), 7
BIC-methods, 7
boot (boot-methods), 8
boot, REBMIX-method (boot-methods), 8
boot, REBMVNORM-method (boot-methods), 8
boot-methods, 8
CAIC (AIC-methods), 4
CAIC, REBMIX-method (AIC-methods), 4
CAIC, REBMVNORM-method (AIC-methods), 4
CAIC-methods (AIC-methods), 4
demix (demix-methods), 11
demix, REBMIX-method (demix-methods), 11
demix-methods, 11
demix-methods, 11
demix-methods, 11
EM.Control-class, 15
PC (PC-methods), 24
PC, REBMIX-method (PC-methods), 24
PC, REBMVNORM-method (PC-methods), 24
PC-methods, 24
ICL (ICL-methods), 18
ICL, REBMIX-method (ICL-methods), 18
ICL, REBMVNORM-method (ICL-methods), 18
ICL-methods, 18
ICLBIC (ICLBIC-methods), 19
iris, 20
kseq, 21, 42, 45
logL, 22
logL, REBMIX-method (logL), 22
logL, REBMVNORM-method (logL), 22
logL-methods (logL), 22
MDL-methods, 23
MDL2 (MDL-methods), 23
MDL2, REBMIX-method (MDL-methods), 23
MDL2, REBMVNORM-method (MDL-methods), 23
MDL2-methods (MDL-methods), 23
MDL5 (MDL-methods), 23
MDL5, REBMIX-method (MDL-methods), 23
MDL5, REBMVNORM-method (MDL-methods), 23
MDL5-methods (MDL-methods), 23
par, 30
plot, RCLRMIX, missing-method (plot-methods), 29
plot, RCLRMVNORM, missing-method (plot-methods), 29
plot, RCLSMIX, missing-method (plot-methods), 29
plot, RCLSMVNORM, missing-method (plot-methods), 29
plot, REBMIX, missing-method (plot-methods), 29
summary, REBMIX-method (REBMIX-methods), 43
summary, REBMIX.boot-method (boot-methods), 8
summary, REBMVNORM-method (REBMIX-methods), 43
summary, REBMVNORM.boot-method (boot-methods), 8

truck, 57

weibull, 58
weibullnormal, 58
wine, 59