

Package ‘semisup’

April 4, 2025

Version 1.31.0

Title Semi-Supervised Mixture Model

Description Implements a parametric semi-supervised mixture model. The permutation test detects markers with main or interactive effects, without distinguishing them. Possible applications include genome-wide association analysis and differential expression analysis.

biocViews SNP, GenomicVariation, SomaticMutation, Genetics,
Classification, Clustering, DNASEq, Microarray,
MultipleComparison

Depends R (>= 3.0.0)

Imports VGAM

Suggests knitr, testthat, SummarizedExperiment

VignetteBuilder knitr

License GPL-3

LazyData true

RoxygenNote 7.0.0

URL <https://github.com/rauschenberger/semisup>

BugReports <https://github.com/rauschenberger/semisup/issues>

git_url <https://git.bioconductor.org/packages/semisup>

git_branch devel

git_last_commit 5bd4eff

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-04-03

Author Armin Rauschenberger [aut, cre]

Maintainer Armin Rauschenberger <armin.rauschenberger@uni.lu>

Contents

semisup-package	2
arguments	3
debug	4
estim.nbinom	5
fit.nbinom	6
fit.norm	7
fit.wrap	8
fit.zinb	9
internal	10
mixture	11
resam.lrts	12
scrutor	13
table	15
toydata	15
Index	16

semisup-package	<i>Semi-supervised mixture model</i>
-----------------	--------------------------------------

Description

This R package implements the semi-supervised mixture model. Use `mixture` for model fitting, and `scrutor` for hypothesis testing.

Getting started

Please type the following commands:

```
utils::vignette("semisup")
?semisup::mixture
?semisup::scrutor
```

More information

A Rauschenberger, RX Menezes, MA van de Wiel, NM van Schoor, and MA Jonker (2020). "Semi-supervised mixture test for detecting markers associated with a quantitative trait", *Manuscript in preparation*.

<a.rauschenberger@vumc.nl>

Description

This page lists and describes all arguments of the R package [semisup](#).

Arguments

<code>y</code>	observations: numeric vector of length <code>n</code>
<code>Y</code>	observations: numeric vector of length <code>n</code> , or numeric matrix with <code>n</code> rows (samples) and <code>q</code> columns (variables)
<code>z</code>	class labels: integer vector of length <code>n</code> , with entries <code>0</code> , <code>1</code> and <code>NA</code>
<code>Z</code>	class labels: numeric vector of length <code>n</code> , or numeric matrix with <code>n</code> rows (samples) and <code>p</code> columns (variables), with entries <code>0</code> and <code>NA</code>
<code>dist</code>	distributional assumption: character "norm" (Gaussian), "nbinom" (negative binomial), or "zinb" (zero-inflated negative binomial)
<code>phi</code>	dispersion parameters: numeric vector of length <code>q</code> , or <code>NULL</code>
<code>pi</code>	zero-inflation parameter(s): numeric vector of length <code>q</code> , or <code>NULL</code>
<code>gamma</code>	offset: numeric vector of length <code>n</code> , or <code>NULL</code>
<code>test</code>	resampling procedure: character "perm" (permutation) or "boot" (parametric bootstrap), or <code>NULL</code>
<code>iter</code>	(maximum) number of resampling iterations : positive integer, or <code>NULL</code>
<code>kind</code>	resampling accuracy: numeric between <code>0</code> and <code>1</code> , or <code>NULL</code> ; all p-values above <code>kind</code> are approximate
<code>starts</code>	restarts of the EM algorithm: positive integer (defaults to <code>1</code>)
<code>it.em</code>	(maximum) number of iterations in the EM algorithm: positive integer (defaults to <code>100</code>)
<code>epsilon</code>	convergence criterion for the EM algorithm: non-negative numeric (defaults to <code>1e-04</code>)
<code>debug</code>	verification of arguments: <code>TRUE</code> or <code>FALSE</code>
<code>pass</code>	parameters for parametric bootstrap algorithm
<code>...</code>	settings EM algorithm: <code>starts</code> , <code>it.em</code> and <code>epsilon</code> (see arguments)

See Also

Use [mixtura](#) for model fitting, and [scrutor](#) for hypothesis testing. All other functions of the R package [semisup](#) are [internal](#).

 debug

Internal function

Description

This function verifies whether the arguments fulfill some formal requirements.

Usage

```
debug(y, z, dist, phi, pi, gamma, test, iter, kind, ...)
```

Arguments

y	observations: numeric vector of length n
z	class labels: integer vector of length n, with entries 0, 1 and NA
dist	distributional assumption: character "norm" (Gaussian), "nbinom" (negative binomial), or "zinb" (zero-inflated negative binomial)
phi	dispersion parameters: numeric vector of length q, or NULL
pi	zero-inflation parameter(s): numeric vector of length q, or NULL
gamma	offset: numeric vector of length n, or NULL
test	resampling procedure: character "perm" (permutation) or "boot" (parametric bootstrap), or NULL
iter	(maximum) number of resampling iterations : positive integer, or NULL
kind	resampling accuracy: numeric between 0 and 1, or NULL; all p-values above kind are approximate
...	settings EM algorithm: starts, it.em and epsilon (see arguments)

Details

If one or more entries of z are equal to 1, the mixture model can be fitted but not tested. Accordingly, kind is replaced by NULL.

Resampling-based testing cannot reach p-values below 1/iter. If kind is smaller than 1/iter, it is replaced by 0.

Value

This function returns warnings and errors. It also returns kind (see details).

See Also

This is an [internal](#) function. The user functions are [mixtura](#) and [scrutor](#).

Examples

```
NULL
```

estim.nbinom	<i>Internal function</i>
--------------	--------------------------

Description

These functions estimate the parameters of the (zero-inflated) negative binomial distribution by applying the maximum likelihood method to the labelled observations in class 0.

Usage

```
estim.nbinom(y, z, gamma)
```

```
estim.zinb(y, z, gamma)
```

Arguments

y	observations: numeric vector of length n
z	class labels: integer vector of length n, with entries 0, 1 and NA
gamma	offset: numeric vector of length n, or NULL

Value

These functions return a list of numerics.

See Also

These are [internal](#) functions. The user functions are [mixtura](#) and [scrutor](#).

Examples

```
# data simulation
n <- 100
y <- stats::rnbino(n=n,mu=5,size=1/0.05)
y[sample(1:n,size=0.2*n)] <- 0
z <- rep(0,times=n)
gamma <- rep(1,times=n)

# parameter estimation
estim.nbinom(y,z,gamma)
estim.zinb(y,z,gamma)
```

 fit.nbinom

Internal function

Description

This function fits the semi-supervised negative binomial mixture model. It is called by [fit.wrap](#).

Usage

```
fit.nbinom(y, z, phi, gamma, it.em, epsilon)
```

Arguments

y	observations: numeric vector of length n
z	class labels: integer vector of length n, with entries 0, 1 and NA
phi	dispersion parameters: numeric vector of length q, or NULL
gamma	offset: numeric vector of length n, or NULL
it.em	(maximum) number of iterations in the EM algorithm: positive integer (defaults to 100)
epsilon	convergence criterion for the EM algorithm: non-negative numeric (defaults to 1e-04)

Value

This function returns the parameter estimates, the posterior probabilities, and the likelihood.

See Also

This is an [internal](#) function. The user functions are [mixtura](#) and [scrutor](#).

Examples

```
# data simulation
n <- 100
z <- rep(0:1, each=n/2)
gamma <- runif(n=n, min=0, max=2)
y <- rnbino(n=n, mu=gamma*(5+2*z), size=1/0.05)
z[(n/4):n] <- NA

# model fitting
fit.nbinom(y, z, phi=0.05, gamma=gamma,
it.em=100, epsilon=1e-04)
```

fit.norm	<i>Internal function</i>
----------	--------------------------

Description

This function fits the semi-supervised Gaussian mixture model. It is called by [fit.wrap](#).

Usage

```
fit.norm(y, z, it.em, epsilon)
```

Arguments

y	observations: numeric vector of length n
z	class labels: integer vector of length n, with entries 0, 1 and NA
it.em	(maximum) number of iterations in the EM algorithm: positive integer (defaults to 100)
epsilon	convergence criterion for the EM algorithm: non-negative numeric (defaults to 1e-04)

Value

This function returns the parameter estimates, the posterior probabilities, and the likelihood.

See Also

This is an [internal](#) function. The user functions are [mixtura](#) and [scrutor](#).

Examples

```
# data simulation
n <- 100
z <- rep(0:1, each=n/2)
y <- rnorm(n=n, mean=2*z, sd=1)
z[(n/4):n] <- NA

# model fitting
fit.norm(y, z, it.em=100, epsilon=1e-04)
```

fit.wrap

*Internal function***Description**

This function fits the semi-supervised mixture model multiple times. It is called by [mixtura](#) and [scrutor](#).

Usage

```
fit.wrap(y, z, dist, phi, pi, gamma, starts = 1, it.em = 100, epsilon = 1e-04)
```

Arguments

y	observations: numeric vector of length n
z	class labels: integer vector of length n, with entries 0, 1 and NA
dist	distributional assumption: character "norm" (Gaussian), "nbinom" (negative binomial), or "zinb" (zero-inflated negative binomial)
phi	dispersion parameters: numeric vector of length q, or NULL
pi	zero-inflation parameter(s): numeric vector of length q, or NULL
gamma	offset: numeric vector of length n, or NULL
starts	restarts of the EM algorithm: positive integer (defaults to 1)
it.em	(maximum) number of iterations in the EM algorithm: positive integer (defaults to 100)
epsilon	convergence criterion for the EM algorithm: non-negative numeric (defaults to 1e-04)

Details

The distributions are parametrised as follows:

- Gaussian
 $y \sim N(\text{mean}, \text{sd}^2)$
 $E[y] = \text{mean}$
 $\text{Var}[y] = \text{sd}^2$
- Negative binomial
 $y \sim \text{NB}(\mu, \text{phi})$
 $E[y] = \mu$
 $\text{Var}[y] = \mu + \text{phi} * \mu^2$
- Zero-inflated negative binomial
 $y \sim \text{ZINB}(\mu, \text{phi}, \text{pi})$
 $E[y] = (1 - \text{pi}) * \mu$

Value

This function returns the parameter estimates, the posterior probabilities, and the likelihood.

posterior	probability of belonging to class 1: numeric vector of length n
converge	path of the log-likelihood: numeric vector with maximum length <code>it.em</code>
estim0	parameter estimates under H0: data frame
estim1	parameter estimates under H1: data frame
loglik0	log-likelihood under H0: numeric
loglik1	log-likelihood under H1: numeric
lrts	likelihood-ratio test statistic: positive numeric

See Also

This is an [internal](#) function. The user functions are [mixture](#) and [scrutor](#).

Examples

```
# data simulation
n <- 100
z <- rep(0:1, each=n/2)
y <- rnorm(n=n, mean=2*z, sd=1)
z[(n/4):n] <- NA

# model fitting
fit.wrap(y, z, dist="norm")
```

fit.zinb

Internal function

Description

This function fits the semi-supervised zero-inflated negative binomial mixture model. It is called by [fit.wrap](#).

Usage

```
fit.zinb(y, z, phi, pi, gamma, it.em, epsilon)
```

Arguments

y	observations: numeric vector of length n
z	class labels: integer vector of length n, with entries 0, 1 and NA
phi	dispersion parameters: numeric vector of length q, or NULL
pi	zero-inflation parameter(s): numeric vector of length q, or NULL

<code>gamma</code>	offset: numeric vector of length <code>n</code> , or <code>NULL</code>
<code>it.em</code>	(maximum) number of iterations in the EM algorithm: positive integer (defaults to 100)
<code>epsilon</code>	convergence criterion for the EM algorithm: non-negative numeric (defaults to $1e-04$)

Value

This function returns the parameter estimates, the posterior probabilities, and the likelihood.

See Also

This is an [internal](#) function. The user functions are [mixture](#) and [scrutor](#).

Examples

```
# data simulation
n <- 100
z <- rep(0:1, each=n/2)
gamma <- runif(n=n, min=0, max=2)
y <- rnbinom(n=n, mu=gamma*(5+2*z), size=1/0.05)
y[sample(1:n, size=0.2*n)] <- 0
z[(n/4):n] <- NA

# model fitting
fit.zinb(y, z, phi=0.05, pi=0.2, gamma=gamma,
it.em=100, epsilon=1e-04)
```

internal

Documentation

Description

This page lists and describes some internal functions of the R package [semisup](#). These functions should not be used for analysing data.

[fit.wrap](#) multiple restarts
[fit.norm](#) Gaussian mixture model
[fit.nbinom](#) negative binomial mixture model
[fit.zinb](#) zero-inflated negative binomial mixture model
[estim.nbinom](#) dispersion estimation
[estim.zinb](#) dispersion and zero-inflation estimation
[resam.lrts](#) resampling (bootstrap, permutation)

See Also

Use [mixture](#) for model fitting, and [scrutor](#) for hypothesis testing.

mixtura	<i>Model fitting</i>
---------	----------------------

Description

This function fits a semi-supervised mixture model. It simultaneously estimates two mixture components, and assigns the unlabelled observations to these.

Usage

```
mixtura(y, z, dist = "norm",
        phi = NULL, pi = NULL, gamma = NULL,
        test = NULL, iter = 100, kind = 0.05,
        debug = TRUE, ...)
```

Arguments

<code>y</code>	observations: numeric vector of length <code>n</code>
<code>z</code>	class labels: integer vector of length <code>n</code> , with entries 0, 1 and NA
<code>dist</code>	distributional assumption: character "norm" (Gaussian), "nbinom" (negative binomial), or "zinb" (zero-inflated negative binomial)
<code>phi</code>	dispersion parameters: numeric vector of length <code>q</code> , or NULL
<code>pi</code>	zero-inflation parameter(s): numeric vector of length <code>q</code> , or NULL
<code>gamma</code>	offset: numeric vector of length <code>n</code> , or NULL
<code>test</code>	resampling procedure: character "perm" (permutation) or "boot" (parametric bootstrap), or NULL
<code>iter</code>	(maximum) number of resampling iterations : positive integer, or NULL
<code>kind</code>	resampling accuracy: numeric between 0 and 1, or NULL; all p-values above <code>kind</code> are approximate
<code>debug</code>	verification of arguments: TRUE or FALSE
<code>...</code>	settings EM algorithm: <code>starts</code> , <code>it.em</code> and <code>epsilon</code> (see arguments)

Details

By default, `phi` and `pi` are estimated by the maximum likelihood method, and `gamma` is replaced by a vector of ones.

Value

This function fits and compares a one-component (H_0) and a two-component (H_1) mixture model.

<code>posterior</code>	probability of belonging to class 1: numeric vector of length <code>n</code>
<code>converge</code>	path of the log-likelihood: numeric vector with maximum length <code>it.em</code>
<code>estim0</code>	parameter estimates under H_0: data frame

estim1	parameter estimates under H1: data frame
loglik0	log-likelihood under H0: numeric
loglik1	log-likelihood under H1: numeric
lrts	likelihood-ratio test statistic: positive numeric
p.value	H0 versus H1: numeric between 0 and 1, or NULL

Reference

A Rauschenberger, RX Menezes, MA van de Wiel, NM van Schoor, and MA Jonker (2020). "Semi-supervised mixture test for detecting markers associated with a quantitative trait", *Manuscript in preparation*.

See Also

Use [scrutor](#) for hypothesis testing. All other functions are [internal](#).

Examples

```
# data simulation
n <- 100
z <- rep(0:1, each=n/2)
y <- rnorm(n=n, mean=2, sd=1)
z[(n/4):n] <- NA

# model fitting
mixtura(y, z, dist="norm", test="perm")
```

resam.lrts

Internal function

Description

This function resamples the data, fits the semi-supervised mixture model, and returns the likelihood ratio test statistic. It is called by [mixtura](#).

Usage

```
resam.lrts(y, z, dist, phi, pi, gamma, test, pass, ...)
```

Arguments

y	observations: numeric vector of length n
z	class labels: integer vector of length n, with entries 0, 1 and NA
dist	distributional assumption: character "norm" (Gaussian), "nbinom" (negative binomial), or "zinb" (zero-inflated negative binomial)

phi	dispersion parameters: numeric vector of length q, or NULL
pi	zero-inflation parameter(s): numeric vector of length q, or NULL
gamma	offset: numeric vector of length n, or NULL
test	resampling procedure: character "perm" (permutation) or "boot" (parametric bootstrap), or NULL
pass	parameters for parametric bootstrap algorithm
...	settings EM algorithm: starts, it.em and epsilon (see arguments)

Value

This function returns a numeric.

See Also

This is an [internal](#) function. The user functions are [mixture](#) and [scrutor](#).

Examples

```
# data simulation
n <- 100
z <- rep(0:1,each=n/2)
y <- rnorm(n=n,mean=2*z,sd=1)
z[(n/4):n] <- NA

# observed test statistic
fit.wrap(y=y,z=z,dist="norm")$lrts

# simulated test statistic
resam.lrts(y=y,z=z,dist="norm",
           phi=NULL,pi=NULL,gamma=NULL,
           test="perm",pass=NULL)
```

scrutor

Hypothesis testing

Description

This function tests whether the unlabelled observations come from a mixture of two distributions.

Usage

```
scrutor(Y, Z, dist = "norm",
        phi = NULL, pi = NULL, gamma = NULL,
        test = "perm", iter = NULL, kind = NULL,
        debug = TRUE, ...)
```

Arguments

Y	observations: numeric vector of length n, or numeric matrix with n rows (samples) and q columns (variables)
Z	class labels: numeric vector of length n, or numeric matrix with n rows (samples) and p columns (variables), with entries 0 and NA
dist	distributional assumption: character "norm" (Gaussian), "nbinom" (negative binomial), or "zinb" (zero-inflated negative binomial)
phi	dispersion parameter(s): numeric vector of length q, or NULL (norm: none, nbinom: MLE)
pi	zero-inflation parameter(s): numeric vector of length q, or NULL (norm: none, nbinom: MLE)
gamma	offset: numeric vector of length n, or NULL
test	resampling procedure: character "perm" (permutation) or "boot" (parametric bootstrap), or NULL
iter	(maximum) number of resampling iterations : positive integer, or NULL
kind	resampling accuracy: numeric between 0 and 1, or NULL; all p-values above kind are approximate
debug	verification of arguments: TRUE or FALSE
...	settings EM algorithm: starts, it.em and epsilon (see arguments)

Details

By default, phi and pi are estimated by the maximum likelihood method, and gamma is replaced by a vector of ones.

Value

This function tests a one-component (H_0) against a two-component mixture model (H_1).

y	index observations
z	index class labels
lrts	test statistic
p.value	p-value

Reference

A Rauschenberger, RX Menezes, MA van de Wiel, NM van Schoor, and MA Jonker (2020). "Semi-supervised mixture test for detecting markers associated with a quantitative trait", *Manuscript in preparation*.

See Also

Use [mixtura](#) for model fitting. All other functions are [internal](#).

Examples

```
# data simulation
n <- 100
z <- rep(0:1,each=n/2)
y <- rnorm(n=n,mean=2*z,sd=1)
z[(n/4):n] <- NA

# hypothesis testing
scrutor(y,z,dist="norm")
```

table	<i>Table</i>
-------	--------------

Description

This dataset includes tables for the approximate mixture test (**not yet available**).

Usage

```
data(table)
```

Format

A list of numeric vectors.

Value

All entries are numeric.

toydata	<i>Toydata</i>
---------	----------------

Description

This dataset allows to reproduce the examples shown in the vignette.

Usage

```
data(toydata)
```

Format

A list of numeric vectors and matrices.

Value

All entries are numeric.

Index

- * **documentation**
 - semisup-package, 2
 - * **internal**
 - arguments, 3
 - debug, 4
 - estim.nbinom, 5
 - fit.nbinom, 6
 - fit.norm, 7
 - fit.wrap, 8
 - fit.zinb, 9
 - internal, 10
 - resam.lrts, 12
 - table, 15
 - toydata, 15
 - * **methods**
 - mixture, 11
 - scrutor, 13
- arguments, 3, 3, 4, 11, 13, 14
- debug, 4
- estim.nbinom, 5, 10
- estim.zinb, 10
- estim.zinb (estim.nbinom), 5
- fit.nbinom, 6, 10
- fit.norm, 7, 10
- fit.wrap, 6, 7, 8, 9, 10
- fit.zinb, 9, 10
- internal, 3–7, 9, 10, 10, 12–14
- mixture, 2–10, 11, 12–14
- resam.lrts, 10, 12
- scrutor, 2–10, 12, 13, 13
- semisup, 3, 10
- semisup (semisup-package), 2
- semisup-package, 2
- table, 15
- toydata, 15