Package ‘semisup’

April 1, 2025

Version 1.30.0

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 RELEASE_3_20

git_last_commit ad0ab69

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2025-03-31

Author Armin Rauschenberger [aut, cre]

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

Contents

`semisup-package` *Semi-supervised mixture model*

Description

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

Getting started

Please type the following commands:

```
utils::vignette("semisup")
?semisup::mixtura
?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>

`arguments` *Documentation*

Description

This page lists and describes all arguments of the R package `semisup`.

Arguments

`y`	**observations:** numeric vector of length `n`
`Y`	**observations:** numeric vector of length `n`, or numeric matrix with `n` rows (samples) and `q` columns (variables)
`z`	**class labels:** integer vector of length `n`, with entries `0`, `1` and `NA`
`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 bionomial), 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
`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`)
`debug`	verification of arguments**:** `TRUE` or `FALSE`
`pass`	parameters for parametric bootstrap algorithm
`...`	settings `EM` algorithm**:** `starts`, `it.em` and `epsilon` (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 bionomial), 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 *Internal function*

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::rnbinom(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 <- rnbinom(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` *Internal function*

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 bionomial), 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 ~ N(mean,sd^2)`
  `E[y]=mean`
  `Var[y]=sd^2`
- Negative binomial
  `y ~ NB(mu,phi)`
  `E[y]=mu`
  `Var[y]=mu+phi*mu^2`

- Zero-inflated negative binomial
  `y ~ ZINB(mu,phi,pi)`
  `E[y]=(1-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 `it.em`
`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 `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.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
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 <- 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 mixtura for model fitting, and scrutor for hypothesis testing.

`mixtura` *Model fitting*

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

`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 bionomial), 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
`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 fits and compares a one-component (`H0`) and a two-component (`H1`) mixture model.

`posterior`	probability of belonging to class `1`: numeric vector of length `n`
`converge`	path of the log-likelihood: numeric vector with maximum length `it.em`
`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
`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 bionomial), 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 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

# 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 bionomial), 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 (H0) against a two-component mixture model (H1).

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 *Table*

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` *Toydata*

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