Package: supclust (via r-universe)

October 15, 2026

Title Supervised Clustering of Predictor Variables Such as Genes

Version 1.1-1

Date 2021-09-25

Description Methodology for supervised grouping aka ``clustering'' of potentially many predictor variables, such as genes etc, implementing algorithms 'PELORA' and 'WILMA'.

Imports graphics, stats, rpart, class

Author Marcel Dettling <marcel.dettling@zhaw.ch> and Martin Maechler

Maintainer Martin Maechler <maechler@stat.math.ethz.ch>

URL https://github.com/mmaechler/supclust

Encoding UTF-8

License GPL-3

NeedsCompilation yes

Repository https://cran.r-universe.dev

Date/Publication 2021-09-27 11:40:02 UTC

RemoteUrl https://github.com/cran/supclust

RemoteRef HEAD

RemoteSha 2bfe21d4797182f3c4004c31939d0c78adc0e59c

Contents

back.search 2
coef.pelora 2
dlda 3
fitted.pelora 4
fitted.wilma 5
leukemia 6
margin 7
pelora 8
plot.pelora 11

plot.wilma 12
predict.pelora 13
predict.wilma 14
print.pelora 16
print.wilma 17
score 18
sign.change 19
sign.flip 20
standardize.genes 21
summary.pelora 22
summary.wilma 23
wilma 23

Index **26**

`back.search` *Internal functions for Supervised Grouping*

Description

These are not to be called by the user.

See Also

`pelora` and `wilma`

`coef.pelora` *Extract the Model Coefficients of Pelora*

Description

Yields the coefficients of the penalized logistic regression model that is fitted by `pelora` with its groups of predictor variables (genes) as input

Usage

```
## S3 method for class 'pelora'
coef(object, ...)
```

Arguments

`object`	an R object of `class` `"pelora"`, typically the result of `pelora()`.
`...`	further arguments passed to and from methods.

Value

A numeric vector of length noc + 1, giving the penalized logistic regression coefficients for the intercept and the noc groups and/or single variables identified by pelora.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

pelora, also for references.

Examples

```
## Running the examples of Pelora's help page
example(pelora, echo = FALSE)
coef(fit)
```

---

dlda *Classification with Wilma's Clusters*

Description

The four functions nnr (nearest neighbor rule), dlda (diagonal linear discriminant analysis), logreg (logistic regression) and aggtrees (aggregated trees) are used for binary classification with the cluster representatives of Wilma's output.

Usage

```
dlda    (xlearn, xtest, ylearn)
nnr     (xlearn, xtest, ylearn)
logreg  (xlearn, xtest, ylearn)
aggtrees(xlearn, xtest, ylearn)
```

Arguments

xlearn	Numeric matrix of explanatory variables (q variables in columns, n cases in rows), containing the learning or training data. Typically, these are the (gene) cluster representatives of Wilma's output.
xtest	A numeric matrix of explanatory variables (q variables in columns, m cases in rows), containing the test or validation data. Typically, these are the fitted (gene) cluster representatives of Wilma's output for the training data, obtained from predict.wilma.
ylearn	Numeric vector of length n containing the class labels for the training observations. These labels have to be coded by 0 and 1.

Details

nnr implements the 1-nearest-neighbor-rule with Euclidean distance function. dlda is linear discriminant analysis, using the restriction that the covariance matrix is diagonal with equal variance for all predictors. logreg is default logistic regression. aggtrees fits a default stump (a classification tree with two terminal nodes) by rpart for every predictor variable and uses majority voting to determine the final classifier.

Value

Numeric vector of length m, containing the predicted class labels for the test observations. The class labels are coded by 0 and 1.

Author(s)

Marcel Dettling

References

see those in wilma.

See Also

wilma

Examples

```
## Generating random learning data: 20 observations and 10 variables (clusters)
set.seed(342)
xlearn <- matrix(rnorm(200), nrow = 20, ncol = 10)

## Generating random test data: 8 observations and 10 variables(clusters)
xtest  <- matrix(rnorm(80),  nrow = 8,  ncol = 10)

## Generating random class labels for the learning data
ylearn <- as.numeric(runif(20)>0.5)

## Predicting the class labels for the test data
nnr(xlearn, xtest, ylearn)
dlda(xlearn, xtest, ylearn)
logreg(xlearn, xtest, ylearn)
aggtrees(xlearn, xtest, ylearn)
```

---

fitted.pelora *Extract the Fitted Values of Pelora*

Description

Yields the fitted values, i.e., the centroids of the (gene) groups that have been identified by pelora.

Usage

```
## S3 method for class 'pelora'
fitted(object, ...)
```

Arguments

object	An R object of class "pelora", typically the result of pelora().
...	Further arguments passed to and from methods.

Value

Numeric matrix of fitted values (for n cases in rows, and noc group centroids in columns).

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

pelora, also for references.

Examples

```
## Running the examples of Pelora's help page
example(pelora, echo = FALSE)
fitted(fit)
```

---

fitted.wilma *Extract the Fitted Values of Wilma*

Description

Yields the fitted values, i.e. the centroids of the (gene) clusters that have been found by wilma.

Usage

```
## S3 method for class 'wilma'
fitted(object, ...)
```

Arguments

object	An R object of class "wilma", typically the result of wilma().
...	further arguments passed to and from methods.

Value

Numeric matrix of fitted values (for n cases in rows, and noc group centroids in columns).

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

wilma, also for references.

Examples

```
## Running the examples of Wilma's help page
example(wilma, echo = FALSE)
fitted(fit)
```

---

leukemia *A part of the Golub's famous AML/ALL-leukemia dataset*

Description

Part of the training set of the famous AML/ALL-leukemia dataset from the Whitehead Institute. It has been reduced to 250 genes, about half of which are very informative for classification, whereas the other half was chosen randomly.

Usage

```
data(leukemia)
```

Format

Contains three R-objects: The expression (38 x 250) matrix `leukemia.x`, the associated binary (`0,1`) response variable `leukemia.y`, and the associated 3-class response variable `leukemia.z` with values in `0,1,2`.

Author(s)

Marcel Dettling

Source

originally at `http://www.genome.wi.mit.edu/MPR/`, (which is not a valid URL any more).

References

First published in
Golub et al. (1999) Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science* **286**, 531–538.

Examples

```
data(leukemia, package="supclust")
str(leukemia.x)
str(leukemia.y)
str(leukemia.z)
op <- par(mfrow= 1:2)
plot(leukemia.x[,56], leukemia.y)
plot(leukemia.x[,174],leukemia.z)
par(op)
```

`margin` *Classification Margin Between Two Sample Classes*

Description

For a set of n observations grouped into two classes (for example n expression values of a gene), the `margin` function measures the size of the gap between the classes. This is the distance between the observation of response class zero having the lowest value, and the individual of with response one having the highest value.

Usage

```
margin(x, resp)
```

Arguments

`x`	Numeric vector of length n, for example containing gene or cluster expression values of n different cases.
`resp`	Numeric vector of length n containing the "binary" class labels of the cases. Must be coded by `0` and `1`.

Value

A numeric value, the `margin`. Positive `margin` indicates perfect separation of the response classes, whereas negative `margin` means imperfect separation.

Author(s)

Marcel Dettling

References

see those in `wilma`.

See Also

`wilma`, `score` is the second statistic that is used there.

Examples

```
data(leukemia, package="supclust")
op <- par(mfrow=c(1,3))
plot(leukemia.x[,69],leukemia.y)
title(paste("Margin = ", round(margin(leukemia.x[,69], leukemia.y),2)))

## Sign-flipping is very important
plot(leukemia.x[,161],leukemia.y)
title(paste("Margin = ", round(margin(leukemia.x[,161], leukemia.y),2)))
x <- sign.flip(leukemia.x, leukemia.y)$flipped.matrix
plot(x[,161],leukemia.y)
title(paste("Margin = ", round(margin(x[,161], leukemia.y),2)))
par(op)
```

---

pelora *Supervised Grouping of Predictor Variables*

Description

Performs selection and supervised grouping of predictor variables in large (microarray gene expression) datasets, with an option for simultaneous classification. Works in a greedy forward strategy and optimizes the binomial log-likelihood, based on estimated conditional probabilities from penalized logistic regression analysis.

Usage

```
pelora(x, y, u = NULL, noc = 10, lambda = 1/32, flip = "pm",
       standardize = TRUE, trace = 1)
```

Arguments

x	Numeric matrix of explanatory variables (p variables in columns, n cases in rows). For example, these can be microarray gene expression data which should be grouped.
y	Numeric vector of length n containing the class labels of the individuals. These labels have to be coded by 0 and 1.
u	Numeric matrix of additional (clinical) explanatory variables (m variables in columns, n cases in rows) that are used in the (penalized logistic regression) prediction model, but neither grouped nor averaged. For example, these can be 'traditional' clinical variables.
noc	Integer, the number of clusters that should be searched for on the data.
lambda	Real, defaults to 1/32. Rescaled penalty parameter that should be in $[0, 1]$.
flip	Character string, describing a method how the x (gene expression) matrix should be sign-flipped. Possible are "pm" (the default) where the sign for each variable is determined upon its entering into the group, "cor" where the sign for each variable is determined a priori as the sign of the empirical correlation of that variable with the y-vector, and "none" where no sign-flipping is carried out.

standardize	Logical, defaults to TRUE. Is indicating whether the predictor variables (genes) should be standardized to zero mean and unit variance.
trace	Integer >= 0; when positive, the output of the internal loops is provided; trace >= 2 provides output even from the internal C routines.

Value

pelora returns an object of class "pelora". The functions print and summary are used to obtain an overview of the variables (genes) that have been selected and the groups that have been formed. The function plot yields a two-dimensional projection into the space of the first two group centroids that pelora found. The generic function fitted returns the fitted values, these are the cluster representatives. coef returns the penalized logistic regression coefficients θ_j for each of the predictors. Finally, predict is used for classifying test data with Pelora's internal penalized logistic regression classifier on the basis of the (gene) groups that have been found.

An object of class "pelora" is a list containing:

genes	A list of length noc, containing integer vectors consisting of the indices (column numbers) of the variables (genes) that have been clustered.
values	A numerical matrix with dimension $n \times$ noc, containing the fitted values, i.e. the group centroids $\tilde{x}_j$.
y	Numeric vector of length n containing the class labels of the individuals. These labels are coded by 0 and 1.
steps	Numerical vector of length noc, showing the number of forward/backward cycles in the fitting process of each cluster.
lambda	The rescaled penalty parameter.
noc	The number of clusters that has been searched for on the data.
px	The number of columns (genes) in the x-matrix.
flip	The method that has been chosen for sign-flipping the x-matrix.
var.type	A factor with noc entries, describing whether the jth predictor is a group of predictors (genes) or a single (clinical) predictor variable.
crit	A list of length noc, containing numerical vectors that provide information about the development of the grouping criterion during the clustering.
signs	Numerical vector of length p, saying whether the ith variable (gene) should be sign-flipped (-1) or not (+1).
samp.names	The names of the samples (rows) in the x-matrix.
gene.names	The names of the variables (columns) in the x-matrix.
call	The function call.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

References

Marcel Dettling (2003) *Finding Predictive Gene Groups from Microarray Data*, see https://stat.ethz.ch/~dettling/supervised.html

Marcel Dettling and Peter Bühlmann (2002). Supervised Clustering of Genes. *Genome Biology*, **3**(12): research0069.1-0069.15, doi: 10.1186/gb2002312research0069.

Marcel Dettling and Peter Bühlmann (2004). Finding Predictive Gene Groups from Microarray Data. *Journal of Multivariate Analysis* **90**, 106–131, doi: 10.1016/j.jmva.2004.02.012

See Also

wilma for another supervised clustering technique.

Examples

```
## Working with a "real" microarray dataset
data(leukemia, package="supclust")

## Generating random test data: 3 observations and 250 variables (genes)
set.seed(724)
xN <- matrix(rnorm(750), nrow = 3, ncol = 250)

## Fitting Pelora
fit <- pelora(leukemia.x, leukemia.y, noc = 3)

## Working with the output
fit
summary(fit)
plot(fit)
fitted(fit)
coef(fit)

## Fitted values and class probabilities for the training data
predict(fit, type = "cla")
predict(fit, type = "prob")

## Predicting fitted values and class labels for the random test data
predict(fit, newdata = xN)
predict(fit, newdata = xN, type = "cla", noc = c(1,2,3))
predict(fit, newdata = xN, type = "pro", noc = c(1,3))

## Fitting Pelora such that the first 70 variables (genes) are not grouped
fit <- pelora(leukemia.x[, -(1:70)], leukemia.y, leukemia.x[,1:70])

## Working with the output
fit
summary(fit)
plot(fit)
fitted(fit)
coef(fit)

## Fitted values and class probabilities for the training data
```

```
predict(fit, type = "cla")
predict(fit, type = "prob")

## Predicting fitted values and class labels for the random test data
predict(fit, newdata = xN[, -(1:70)], newclin = xN[, 1:70])
predict(fit, newdata = xN[, -(1:70)], newclin = xN[, 1:70], "cla", noc  = 1:10)
predict(fit, newdata = xN[, -(1:70)], newclin = xN[, 1:70], type = "pro")
```

`plot.pelora` *2-Dimensional Visualization of Pelora's Output*

Description

Yields a projection of the cases (for example n gene expression profiles) into the space of the first two gene group centroids that were identified by `pelora`.

Usage

```
## S3 method for class 'pelora'
plot(x, main = "2-Dimensional Projection Pelora's output",
            xlab = NULL, ylab = NULL, col = seq(x$yvals), ...)
```

Arguments

`x`	An R object of `class` `"pelora"`, typically the result of `pelora()`.
`main`	A character string, giving the title of the plot.
`xlab`	A character string, giving the annotation of the x-axis.
`ylab`	A character string, giving the annotation of the x-axis.
`col`	A numeric vector of length 2, coding the colors that will be used for plotting the class labels.
`...`	Further arguments passed to and from methods.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

`pelora`, also for references.

Examples

```
## Running the examples of Pelora's help page
example(pelora, echo = FALSE)
plot(fit)
```

plot.wilma *2-Dimensional Visualization of Wilma's Output*

Description

Yields a projection of the cases (for example n gene expression profiles) into the space of the first two gene group centroids that were identified by wilma.

Usage

```
## S3 method for class 'wilma'
plot(x, xlab = NULL, ylab = NULL, col = seq(x$yvals),
           main = "2-Dimensional Projection of Wilma's Output", ...)
```

Arguments

x	an R object of class "wilma", typically the result of wilma().
xlab	character string, giving the annotation of the x-axis.
ylab	character string, giving the annotation of the x-axis.
col	a numeric vector of length 2, coding the colors that will be used for plotting the class labels.
main	a character string, giving the title of the plot.
...	Further arguments passed to and from methods.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

wilma, also for references.

Examples

```
## Running the examples of Wilma's help page
example(wilma, echo = FALSE)
plot(fit)
```

predict.pelora *Predict Method for Pelora*

Description

Yields fitted values, predicted class labels and conditional probability estimates for training and test data, which are based on the gene groups pelora found, and on its internal penalized logistic regression classifier.

Usage

```
## S3 method for class 'pelora'
predict(object, newdata = NULL, newclin = NULL,
               type = c("fitted", "probs", "class"), noc = object$noc, ...)
```

Arguments

object	An R object of class "pelora", typically the result of pelora().
newdata	Numeric matrix with the same number of explanatory variables as the original x-matrix (p variables in columns, r cases in rows). For example, these can be additional microarray gene expression data which should be predicted.
newclin	Numeric matrix with the same number of additional (clinical) explanatory variables as the original u-matrix (m variables in columns, r cases in rows) that are used in the (penalized logistic regression) prediction model, but neither grouped nor averaged. Only needs to be given, if the model fit included an u-matrix. For example, these can be 'traditional' clinical variables.
type	Character string, describing whether fitted values "fitted", estimated conditional probabilites "probs" or class labels "class" should be returned.
noc	Integer, saying with how many clusters the fitted values, probability estimates or class labels should be determined. Also numeric vectors are allowed as an argument. The output is then a numeric matrix with fitted values, probability estimates or class labels for a multiple number of clusters.
...	Further arguments passed to and from methods.

Details

If newdata = NULL, then the in-sample fitted values, probability estimates and class label predictions are returned.

Value

Depending on whether noc is a single number or a numeric vector. In the first case, a numeric vector of length r is returned, which contains fitted values for noc clusters, or probability estimates/class label predictions with noc clusters.

In the latter case, a numeric matrix with length(noc) columns, each containing fitted values for noc clusters, or probability estimates/class label predictions with noc clusters, is returned.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

pelora, also for references.

Examples

```
## Working with a "real" microarray dataset
data(leukemia, package="supclust")

## Generating random test data: 3 observations and 250 variables (genes)
set.seed(724)
xN <- matrix(rnorm(750), nrow = 3, ncol = 250)

## Fitting Pelora
fit <- pelora(leukemia.x, leukemia.y, noc = 3)

## Fitted values and class probabilities for the training data
predict(fit, type = "cla")
predict(fit, type = "prob")

## Predicting fitted values and class labels for the random test data
predict(fit, newdata = xN)
predict(fit, newdata = xN, type = "cla", noc = c(1,2,3))
predict(fit, newdata = xN, type = "pro", noc = c(1,3))

## Fitting Pelora such that the first 70 variables (genes) are not grouped
fit <- pelora(leukemia.x[, -(1:70)], leukemia.y, leukemia.x[,1:70])

## Fitted values and class probabilities for the training data
predict(fit, type = "cla")
predict(fit, type = "prob")

## Predicting fitted values and class labels for the random test data
predict(fit, newdata = xN[, -(1:70)], newclin = xN[, 1:70])
predict(fit, newdata = xN[, -(1:70)], newclin = xN[, 1:70], "cla", noc  = 1:10)
predict(fit, newdata = xN[, -(1:70)], newclin = xN[, 1:70], type = "pro")
```

---

predict.wilma *Predict Method for Wilma*

Description

Yields fitted values or predicted class labels for training and test data, which are based on the supervised gene clusters wilma found, and on a choice of four different classifiers: the nearest-neighbor rule, diagonal linear discriminant analysis, logistic regression and aggregated trees.

Usage

```
## S3 method for class 'wilma'
predict(object, newdata = NULL, type = c("fitted", "class"),
              classifier = c("nnr", "dlda", "logreg", "aggtrees"),
              noc = object$noc, ...)
```

Arguments

object	an R object of class "wilma", typically the result of wilma().
newdata	numeric matrix with the same number of explanatory variables as the original x-matrix (p variables in columns, r cases in rows). For example, these can be additional microarray gene expression data which should be predicted.
type	character string describing whether fitted values "fitted" or predicted class labels "class" should be returned.
classifier	character string specifying which classifier should be used. Choices are "nnr", the 1-nearest-neighbor-rule; "dlda", diagonal linear discriminant analysis; "logreg", logistic regression; "aggtrees" aggregated trees.
noc	integer specifying how many clusters the fitted values or class label predictions should be determined. Also numeric vectors are allowed as an argument. The output is then a numeric matrix with fitted values or class label predictions for a multiple number of clusters.
...	further arguments passed to and from methods.

Details

If newdata = NULL, then the in-sample fitted values or class label predictions are returned.

Value

Depending on whether noc is a single number or a numeric vector. In the first case, a numeric vector of length r is returned, which contains fitted values for noc clusters, or class label predictions with noc clusters.

In the latter case, a numeric matrix with length(noc) columns, each containing fitted values for noc clusters, or class label predictions with noc clusters, is returned.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

wilma also for *references*, and for the four classifiers nnr, dlda, logreg, aggtrees.

Examples

```
## Working with a "real" microarray dataset
data(leukemia, package="supclust")

## Generating random test data: 3 observations and 250 variables (genes)
set.seed(724)
xN <- matrix(rnorm(750), nrow = 3, ncol = 250)

## Fitting Wilma
fit  <- wilma(leukemia.x, leukemia.y, noc = 3, trace = 1)

## Fitted values and class predictions for the training data
predict(fit, type = "cla")
predict(fit, type = "fitt")

## Predicting fitted values and class labels for test data
predict(fit, newdata = xN)
predict(fit, newdata = xN, type = "cla", classifier = "nnr", noc = c(1,2,3))
predict(fit, newdata = xN, type = "cla", classifier = "dlda", noc = c(1,3))
predict(fit, newdata = xN, type = "cla", classifier = "logreg")
predict(fit, newdata = xN, type = "cla", classifier = "aggtrees")
```

`print.pelora` *Print Method for Pelora Objects*

Description

Yields an overview about the type, size and final criterion value of the predictor variables that were selected by `pelora`.

Usage

```
## S3 method for class 'pelora'
print(x, digits = getOption("digits"), details = FALSE, ...)
```

Arguments

`x`	an R object of `class` `"pelora"`, typically the result of `pelora()`.
`digits`	the number of digits that should be printed.
`details`	logical, defaults to `FALSE`. If set to `TRUE`, the output corresponds to `summary.pelora`.
`...`	Further arguments passed to and from methods.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

`pelora`, also for references.

Examples

```
## Running the examples of Pelora's help page
example(pelora, echo = FALSE)
print(fit)
```

`print.wilma` *Print Method for Wilma Objects*

Description

Yields an overview about the size and the final criterion values of the clusters that were selected by `wilma`.

Usage

```
## S3 method for class 'wilma'
print(x, ...)
```

Arguments

`x`	An R object of `class` `"wilma"`, typically the result of `wilma()`.
`...`	Further arguments passed to and from methods.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

`wilma`, also for references.

Examples

```
## Running the examples of Wilma's help page
example(wilma, echo = FALSE)
print(fit)
```

score *Wilcoxon Score for Binary Problems*

Description

For a set of n observations grouped into two classes (for example n expression values of a gene), the score function measures the separation of the classes. It can be interpreted as counting for each observation having response zero, the number of individuals of response class one that are smaller, and summing up these quantities.

Usage

```
score(x, resp)
```

Arguments

x	Numeric vector of length n, for example containing gene or cluster expression values of n different cases.
resp	Numeric vector of length n containing the "binary" class labels of the cases. Must be coded by 0 and 1.

Value

A numeric value, the score. The minimal score is zero, the maximal score is the product of the number of samples in class 0 and class 1. Values near the minimal or maximal score indicate good separation, whereas intermediate score means poor separation.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

wilma also for *references*; margin is the second statistic that is used there.

Examples

```
data(leukemia, package="supclust")
op <- par(mfrow=c(1,3))
plot(leukemia.x[,69],leukemia.y)
title(paste("Score = ", score(leukemia.x[,69], leukemia.y)))

## Sign-flipping is very important
plot(leukemia.x[,161],leukemia.y)
title(paste("Score = ", score(leukemia.x[,161], leukemia.y),2))
x <- sign.flip(leukemia.x, leukemia.y)$flipped.matrix
plot(x[,161],leukemia.y)
title(paste("Score = ", score(x[,161], leukemia.y),2))
par(op)
```

`sign.change` *Sign-flipping of Predictor Variables to Obtain Equal Polarity*

Description

Computes the empirical correlation for each predictor variable (gene) in the `x`-Matrix with the response `y`, and multiplies its values with (-1) if the empirical correlation has a negative sign. For gene expression data, this amounts to treating under- and overexpression symmetrically. After the `sign.change`, low (expression) values point towards response class 0 and high (expression) values point towards class 1.

Usage

```
sign.change(x, y)
```

Arguments

`x`	Numeric matrix of explanatory variables (p variables in columns, n cases in rows). For example, these can be microarray gene expression data which should be sign-flipped and then grouped.
`y`	Numeric vector of length n containing the class labels of the individuals. These labels have to be coded by 0 and 1.

Value

Returns a list containing:

`x.new`	The sign-flipped `x`-matrix.
`signs`	Numeric vector of length p, which for each predictor variable indicates whether it was sign-flipped (coded by -1) or not (coded by +1).

Author(s)

Marcel Dettling, `<dettling@stat.math.ethz.ch>`

See Also

`pelora` also for *references*, as well as for older methodology, `wilma` and `sign.flip`.

Examples

```
data(leukemia, package="supclust")

op <- par(mfrow=c(1,3))
plot(leukemia.x[,69],leukemia.y)
title(paste("Margin = ", round(margin(leukemia.x[,69], leukemia.y),2)))

## Sign-flipping is very important
```

```
plot(leukemia.x[,161],leukemia.y)
title(paste("Margin = ", round(margin(leukemia.x[,161], leukemia.y),2)))
x <- sign.change(leukemia.x, leukemia.y)$x.new
plot(x[,161],leukemia.y)
title(paste("Margin = ", round(margin(x[,161], leukemia.y),2)))
par(op)
```

`sign.flip` *Sign-flipping of Predictor Variables to Obtain Equal Polarity*

Description

Computes the `score` for each predictor variable (gene) in the `x`-Matrix, and multiplies its values with (-1) if its `score` is greater or equal than half of the maximal `score`. For gene expression data, this amounts to treating under- and overexpression symmetrically. After the sign-flip procedure, low (expression) values point towards response class 0 and high (expression) values point towards class 1.

Usage

```
sign.flip(x, y)
```

Arguments

`x`	Numeric matrix of explanatory variables (p variables in columns, n cases in rows). For example, these can be microarray gene expression data which should be sign-flipped and then clustered.
`y`	Numeric vector of length n containing the class labels of the individuals. These labels have to be coded by 0 and 1.

Value

Returns a list containing:

`flipped.matrix`	The sign-flipped `x`-matrix.
`signs`	Numeric vector of length p, which for each predictor variable indicates whether it was sign-flipped (coded by -1) or not (coded by +1).

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

`wilma` also for the *references* and `score`, as well as for a newer methodology, `pelora` and `sign.change`.

Examples

```
data(leukemia, package="supclust")

op <- par(mfrow=c(1,3))
plot(leukemia.x[,69],leukemia.y)
title(paste("Margin = ", round(margin(leukemia.x[,69], leukemia.y),2)))

## Sign-flipping is very important
plot(leukemia.x[,161],leukemia.y)
title(paste("Margin = ", round(margin(leukemia.x[,161], leukemia.y),2)))
x <- sign.flip(leukemia.x, leukemia.y)$flipped.matrix
plot(x[,161],leukemia.y)
title(paste("Margin = ", round(margin(x[,161], leukemia.y),2)))
par(op)# reset
```

---

standardize.genes *Standardization of Predictor Variables*

Description

Standardizes each column (gene) of the x-matrix to zero mean and unit variance. This function is not to be called by the user, the standardization is handled internally in pelora.

Usage

```
standardize.genes(exmat)
```

Arguments

exmat	Numeric matrix of explanatory variables (p variables in columns, n cases in rows). For example, these can be microarray gene expression data which should be standardized and then grouped.

Value

Returns a list containing:

x	The standardized x-matrix
means	Numeric vector of length p, containing the initial mean of each column (gene) of the x-matrix.
sdevs	Numeric vector of length p, containing the initial standard deviation of each column (gene) of the x-matrix.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

pelora also for the *references*.

---

summary.pelora *Summary Method for Pelora Objects*

Description

Yields detailed information about the variables (genes) that have been selected, and how they were grouped.

Usage

```
## S3 method for class 'pelora'
summary(object, digits, ...)
```

Arguments

object	an R object of class "pelora", typically the result of pelora().
digits	The number of digits that should be printed.
...	Further arguments passed to and from methods.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

pelora, also for references.

Examples

```
## Running the examples of Pelora's help page
example(pelora, echo = FALSE)
summary(fit)
```

summary.wilma *Summary Method for Wilma Objects*

Description

Yields detailed information about the variables (genes) that have been selected, and how they were clustered.

Usage

```
## S3 method for class 'wilma'
summary(object, ...)
```

Arguments

object	An R object of class "wilma", typically the result of wilma().
...	Further arguments passed to and from methods.

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

See Also

wilma, also for references.

Examples

```
## Running the examples of Wilma's help page
example(wilma, echo = FALSE)
summary(fit)
```

wilma *Supervised Clustering of Predictor Variables*

Description

Performs supervised clustering of predictor variables for large (microarray gene expression) datasets. Works in a greedy forward strategy and optimizes a combination of the Wilcoxon and Margin statistics for finding the clusters.

Usage

```
wilma(x, y, noc, genes = NULL, flip = TRUE, once.per.clust = FALSE, trace = 0)
```

Arguments

x	Numeric matrix of explanatory variables (p variables in columns, n cases in rows). For example, these can be microarray gene expression data which should be clustered.
y	Numeric vector of length n containing the class labels of the individuals. These labels have to be coded by 0 and 1.
noc	Integer, the number of clusters that should be searched for on the data.
genes	Defaults to NULL. An optional list (of length noc) of vectors containing the indices (column numbers) of the previously known initial clusters.
flip	Logical, defaults to TRUE. Is indicating whether the clustering should be done with or without sign-flipping.
once.per.clust	Logical, defaults to FALSE. Is indicating if each variable (gene) should only be allowed to enter into each cluster once; equivalently, the cluster mean profile has only weights ± 1 for each variable.
trace	Integer >= 0; when positive, the output of the internal loops is provided; trace >= 2 provides output even from the internal C routines.

Value

wilma returns an object of class "wilma". The functions print and summary are used to obtain an overview of the clusters that have been found. The function plot yields a two-dimensional projection into the space of the first two clusters that wilma found. The generic function fitted returns the fitted values, these are the cluster representatives. Finally, predict is used for classifying test data on the basis of Wilma's cluster with either the nearest-neighbor-rule, diagonal linear discriminant analysis, logistic regression or aggregated trees.

An object of class "wilma" is a list containing:

clist	A list of length noc, containing integer vectors consisting of the indices (column numbers) of the variables (genes) that have been clustered.
steps	Numerical vector of length noc, showing the number of forward/backward cycles in the fitting process of each cluster.
y	Numeric vector of length n containing the class labels of the individuals. These labels have to be coded by 0 and 1.
x.means	A list of length noc, containing numerical matrices consisting of the cluster representatives after insertion of each variable.
noc	Integer, the number of clusters that has been searched for on the data.
signs	Numerical vector of length p, saying whether the ith variable (gene) should be sign-flipped (-1) or not (+1).

Author(s)

Marcel Dettling, <dettling@stat.math.ethz.ch>

References

Marcel Dettling (2002) *Supervised Clustering of Genes*, see https://stat.ethz.ch/~dettling/supercluster.html

Marcel Dettling and Peter Bühlmann (2002). Supervised Clustering of Genes. *Genome Biology*, **3**(12): research0069.1-0069.15, doi: 10.1186/gb2002312research0069 .

Marcel Dettling and Peter Bühlmann (2004). Finding Predictive Gene Groups from Microarray Data. *Journal of Multivariate Analysis* **90**, 106–131, doi: 10.1016/j.jmva.2004.02.012 .

See Also

score, margin, and for a newer methodology, pelora.

Examples

```
## Working with a "real" microarray dataset
data(leukemia, package="supclust")

## Generating random test data: 3 observations and 250 variables (genes)
set.seed(724)
xN <- matrix(rnorm(750), nrow = 3, ncol = 250)

## Fitting Wilma
fit  <- wilma(leukemia.x, leukemia.y, noc = 3, trace = 1)

## Working with the output
fit
summary(fit)
plot(fit)
fitted(fit)

## Fitted values and class predictions for the training data
predict(fit, type = "cla")
predict(fit, type = "fitt")

## Predicting fitted values and class labels for test data
predict(fit, newdata = xN)
predict(fit, newdata = xN, type = "cla", classifier = "nnr", noc = c(1,2,3))
predict(fit, newdata = xN, type = "cla", classifier = "dlda", noc = c(1,3))
predict(fit, newdata = xN, type = "cla", classifier = "logreg")
predict(fit, newdata = xN, type = "cla", classifier = "aggtrees")
```

Index

∗ **classif**
  back.search, 2
  coef.pelora, 2
  dlda, 3
  fitted.pelora, 4
  fitted.wilma, 5
  pelora, 8
  plot.pelora, 11
  plot.wilma, 12
  predict.pelora, 13
  predict.wilma, 14
  print.pelora, 16
  print.wilma, 17
  summary.pelora, 22
  summary.wilma, 23
∗ **cluster**
  back.search, 2
  coef.pelora, 2
  fitted.pelora, 4
  fitted.wilma, 5
  pelora, 8
  plot.pelora, 11
  plot.wilma, 12
  predict.pelora, 13
  predict.wilma, 14
  print.pelora, 16
  print.wilma, 17
  summary.pelora, 22
  summary.wilma, 23
  wilma, 23
∗ **datasets**
  leukemia, 6
∗ **htest**
  margin, 7
  score, 18
∗ **manip**
  sign.change, 19
  sign.flip, 20
  standardize.genes, 21

aggtrees, *15*
aggtrees *(*dlda*)*, 3

back.search, 2

class, *2*, *5*, *11–13*, *15–17*, *22*, *23*
coef.pelora, 2

dlda, 3, *15*

fitted.pelora, 4
fitted.wilma, 5

leukemia, 6
logreg, *15*
logreg *(*dlda*)*, 3

margin, 7, *18*, *25*

nnr, *15*
nnr *(*dlda*)*, 3

p.1clust *(*back.search*)*, 2
pelora, *2–5*, 8, *11*, *13*, *14*, *16*, *19*, *20*, *22*, *25*
plot.pelora, 11
plot.wilma, 12
predict.pelora, 13
predict.wilma, 14
print.pelora, 16
print.wilma, 17
printClist *(*back.search*)*, 2

ridge.coef *(*back.search*)*, 2

score, *7*, 18, *20*, *25*
sign.change, 19, *20*
sign.flip, *19*, 20
standardize.genes, 21
summary.pelora, *16*, 22
summary.wilma, 23

wilma, *2*, *4–7*, *10*, *12*, *15*, *17–20*, 23, *23*