Package ‘plsVarSel’

July 23, 2025

Encoding UTF-8

Type Package

Title Variable Selection in Partial Least Squares

Version 0.9.13

Date 2025-04-09

Description Interfaces and methods for variable selection in Partial Least
Squares. The methods include filter methods, wrapper methods and embedded
methods. Both regression and classification is supported.

License GPL (>= 2)

URL `https://github.com/khliland/plsVarSel/`

BugReports `https://github.com/khliland/plsVarSel/issues/`

Depends pls

Imports grDevices, graphics, genalg, mvtnorm, bdsmatrix, MASS,
progress, parallel, stats, praznik

Suggests Rmpi

RoxygenNote 7.3.2

NeedsCompilation no

Author Kristian Hovde Liland [aut, cre] (ORCID:
<`https://orcid.org/0000-0001-6468-9423`>),
Tahir Mehmood [ctb],
Solve Sæbø [ctb]

Maintainer Kristian Hovde Liland <`kristian.liland@nmbu.no`>

Repository CRAN

Date/Publication 2025-04-09 08:00:02 UTC

Contents

`bve_pls` *Backward variable elimination PLS (BVE-PLS)*

Description

A backward variable elimination procedure for elimination of non informative variables.

Usage

```
bve_pls(y, X, ncomp = 10, ratio = 0.75, VIP.threshold = 1)
```

Arguments

`y`	vector of response values (`numeric` or `factor`).
`X`	numeric predictor `matrix`.
`ncomp`	integer number of components (default = 10).
`ratio`	the proportion of the samples to use for calibration (default = 0.75).
`VIP.threshold`	thresholding to remove non-important variables (default = 1).

Details

Variables are first sorted with respect to some importancemeasure, and usually one of the filter measures described above are used. Secondly, a threshold is used to eliminate a subset of the least informative variables. Then a model is fitted again to the remaining variables and performance is measured. The procedure is repeated until maximum model performance is achieved.

Value

Returns a vector of variable numbers corresponding to the model having lowest prediction error.

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

I. Frank, Intermediate least squares regression method, Chemometrics and Intelligent Laboratory Systems 1 (3) (1987) 233-242.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(gasoline, package = "pls")
with( gasoline, bve_pls(octane, NIR) )
```

`covSel` *Covariance Selection - CovSel*

Description

Sequential selection of variables based on squared covariance with response and intermediate deflation (as in Partial Least Squares).

Usage

```
covSel(X, Y, nvar)
```

Arguments

`X`	`matrix` of input variables
`Y`	`matrix` of response variable(s)
`nvar`	maximum number of variables

Value

`selected`	an integer vector of selected variables
`scores`	a matrix of score vectors
`loadings`	a matrix of loading vectors
`Yloadings`	a matrix of Y loadings

References

J.M. Roger, B. Palagos, D. Bertrand, E. Fernandez-Ahumada. CovSel: Variable selection for highly multivariate and multi-response calibration: Application to IR spectroscopy. Chemom Intel Lab Syst. 2011;106(2):216-223. P. Mishra, A brief note on a new faster covariate's selection (fCovSel) algorithm, Journal of Chemometrics 36(5) 2022.

Examples

```
data(gasoline, package = "pls")
sels <- with(gasoline, covSel(NIR, octane, 5))
matplot(t(gasoline$NIR), type = "l")
abline(v = sels$selected, col = 2)
```

---

filterPLSR *Optimisation of filters for Partial Least Squares*

Description

Extract the index of influential variables based on threshold defiend for LW (loading weights), RC (regression coef), JT (jackknife testing) and VIP (variable importance on projection).

Usage

```
filterPLSR(
  y,
  X,
  ncomp = 10,
  ncomp.opt = c("minimum", "same"),
  validation = "LOO",
  LW.threshold = NULL,
  RC.threshold = NULL,
  URC.threshold = NULL,
  FRC.threshold = NULL,
  JT.threshold = NULL,
  VIP.threshold = NULL,
  SR.threshold = NULL,
  sMC.threshold = NULL,
  mRMR.threshold = NULL,
  WVC.threshold = NULL,
  ...
)
```

Arguments

`y`	vector of response values (`numeric` or `factor`).
`X`	numeric predictor `matrix`.
`ncomp`	integer number of components (default = 10).

`ncomp.opt`	use the number of components corresponding to minimum error (minimum) or `ncomp` (same).
`validation`	type of validation in the PLS modelling (default = "LOO").
`LW.threshold`	threshold for Loading Weights if applied (default = NULL).
`RC.threshold`	threshold for Regression Coefficients if applied (default = NULL).
`URC.threshold`	threshold for Unit normalized Regression Coefficients if applied (default = NULL).
`FRC.threshold`	threshold for Fitness normalized Regression Coefficients if applied (default = NULL).
`JT.threshold`	threshold for Jackknife Testing if applied (default = NULL).
`VIP.threshold`	threshold for Variable Importance on Projections if applied (default = NULL).
`SR.threshold`	threshold for Selectivity Ration if applied (default = NULL).
`sMC.threshold`	threshold for Significance Multivariate Correlation if applied (default = NULL).
`mRMR.threshold`	threshold for minimum Redundancy Maximum Releveance if applied (default = NULL).
`WVC.threshold`	threshold for Weighted Variable Contribution if applied (default = NULL).
`...`	additional paramters for `pls`, e.g. segmentation or similar.

Details

Filter methods are applied for variable selection with PLSR. This function can return selected variables and Root Mean Squared Error of Cross-Validation for various filter methods and determine optimum numbers of components.

Value

Returns a list of lists containing filters (outer list), their selected variables, optimal numbers of components and prediction accuracies.

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

T. Mehmood, K.H. Liland, L. Snipen, S. Sæbø, A review of variable selection methods in Partial Least Squares Regression, Chemometrics and Intelligent Laboratory Systems 118 (2012) 62-69.

See Also

`VIP` (SR/sMC/LW/RC/URC/FRC/mRMR), `filterPLSR`, `spa_pls`, `stpls`, `truncation`, `bve_pls`, `mcuve_pls`, `ipw_pls`, `ga_pls`, `rep_pls`, `WVC_pls`, `T2_pls`.

Examples

```
data(gasoline, package = "pls")
## Not run:
with( gasoline, filterPLSR(octane, NIR, ncomp = 10, "minimum", validation = "LOO",
 RC.threshold = c(0.1,0.5), SR.threshold = 0.5))

## End(Not run)
```

`ga_pls` *Genetic algorithm combined with PLS regression (GA-PLS)*

Description

A subset search algorithm inspired by biological evolution theory and natural selection.

Usage

```
ga_pls(y, X, GA.threshold = 10, iters = 5, popSize = 100)
```

Arguments

`y`	vector of response values (`numeric` or `factor`).
`X`	numeric predictor `matrix`.
`GA.threshold`	the change for a zero for mutations and initialization (default = 10). (The ratio of non-selected variables for each chromosome.)
`iters`	the number of iterations (default = 5).
`popSize`	the population size (default = 100).

Details

1. Building an initial population of variable sets by setting bits for each variable randomly, where bit '1' represents selection of corresponding variable while '0' presents non-selection. The approximate size of the variable sets must be set in advance.

2. Fitting a PLSR-model to each variable set and computing the performance by, for instance, a leave one out cross-validation procedure.

3. A collection of variable sets with higher performance are selected to survive until the next "generation".

4. Crossover and mutation: new variable sets are formed 1) by crossover of selected variables between the surviving variable sets, and 2) by changing (mutating) the bit value for each variable by small probability.

5. The surviving and modified variable sets form the population serving as input to point 2.

Value

Returns a vector of variable numbers corresponding to the model having lowest prediction error.

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

K. Hasegawa, Y. Miyashita, K. Funatsu, GA strategy for variable selection in QSAR studies: GA-based PLS analysis of calcium channel antagonists, Journal of Chemical Information and Computer Sciences 37 (1997) 306-310.

See Also

VIP (SR/sMC/LW/RC), filterPLSR, shaving, stpls, truncation, bve_pls, ga_pls, ipw_pls, mcuve_pls, rep_pls, spa_pls, lda_from_pls, lda_from_pls_cv, setDA.

Examples

```
data(gasoline, package = "pls")
# with( gasoline, ga_pls(octane, NIR, GA.threshold = 10) ) # Time-consuming
```

---

ipw_pls *Iterative predictor weighting PLS (IPW-PLS)*

Description

An iterative procedure for variable elimination.

Usage

```
ipw_pls(
  y,
  X,
  ncomp = 10,
  no.iter = 10,
  IPW.threshold = 0.01,
  filter = "RC",
  scale = TRUE
)

ipw_pls_legacy(y, X, ncomp = 10, no.iter = 10, IPW.threshold = 0.1)
```

Arguments

y	vector of response values (numeric or factor).
X	numeric predictor matrix.
ncomp	integer number of components (default = 10).
no.iter	the number of iterations (default = 10).

`IPW.threshold`	threshold for regression coefficients (default = 0.1).
`filter`	which filtering method to use (among "RC", "SR", "LW", "VIP", "sMC")
`scale`	standardize data (default=TRUE, as in reference)

Details

This is an iterative elimination procedure where a measure of predictor importance is computed after fitting a PLSR model (with complexity chosen based on predictive performance). The importance measure is used both to re-scale the original X-variables and to eliminate the least important variables before subsequent model re-fitting

The IPW implementation was corrected in `plsVarSel` version 0.9.5. For backward compatibility the old implementation is included as `ipw_pls_legacy`.

Value

Returns a vector of variable numbers corresponding to the model having lowest prediction error.

Author(s)

Kristian Hovde Liland

References

M. Forina, C. Casolino, C. Pizarro Millan, Iterative predictor weighting (IPW) PLS: a technique for the elimination of useless predictors in regression problems, Journal of Chemometrics 13 (1999) 165-184.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `setDA`.

Examples

```
data(gasoline, package = "pls")
with( gasoline, ipw_pls(octane, NIR) )
```

`lda_from_pls` *LDA/QDA classification from PLS model*

Description

For each number of components LDA/QDA models are created from the scores of the supplied PLS model and classifications are performed.

Usage

```
lda_from_pls(model, grouping, newdata, ncomp)
```

Arguments

`model`	pls model fitted with the `pls` package
`grouping`	vector of grouping labels
`newdata`	predictors in the same format as in the `pls` model
`ncomp`	maximum number of PLS components

Value

matrix of classifications

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(mayonnaise, package = "pls")
mayonnaise <- within(mayonnaise, {dummy <- model.matrix(~y-1,data.frame(y=factor(oil.type)))})
pls <- plsr(dummy ~ NIR, ncomp = 10, data = mayonnaise, subset = train)
with(mayonnaise, {
 classes <- lda_from_pls(pls, oil.type[train], NIR[!train,], 10)
 colSums(oil.type[!train] == classes) # Number of correctly classified out of 42
})
```

`lda_from_pls_cv` *Cross-validated LDA/QDA classification from PLS model*

Description

For each number of components LDA/QDA models are created from the scores of the supplied PLS model and classifications are performed. This use of cross-validation has limitations. Handle with care!

Usage

```
lda_from_pls_cv(model, X, y, ncomp, Y.add = NULL)
```

Arguments

`model`	`pls` model fitted with the `pls` package
`X`	predictors in the same format as in the `pls` model
`y`	vector of grouping labels
`ncomp`	maximum number of PLS components
`Y.add`	additional responses

Value

matrix of classifications

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(mayonnaise, package = "pls")
mayonnaise <- within(mayonnaise, {dummy <- model.matrix(~y-1,data.frame(y=factor(oil.type)))})
pls <- plsr(dummy ~ NIR, ncomp = 8, data = mayonnaise, subset = train,
            validation = "CV", segments = 40, segment.type = "consecutive")
with(mayonnaise, {
 classes <- lda_from_pls_cv(pls, NIR[train,], oil.type[train], 8)
 colSums(oil.type[train] == classes) # Number of correctly classified out of 120
})
```

`mcuve_pls` *Uninformative variable elimination in PLS (UVE-PLS)*

Description

Artificial noise variables are added to the predictor set before the PLSR model is fitted. All the original variables having lower "importance" than the artificial noise variables are eliminated before the procedure is repeated until a stop criterion is reached.

Usage

```
mcuve_pls(y, X, ncomp = 10, N = 3, ratio = 0.75, MCUVE.threshold = NA)
```

Arguments

`y`	vector of response values (`numeric` or `factor`).
`X`	numeric predictor `matrix`.
`ncomp`	integer number of components (default = 10).
`N`	number of samples Mone Carlo simulations (default = 3).
`ratio`	the proportion of the samples to use for calibration (default = 0.75).
`MCUVE.threshold`	thresholding separate signal from noise (default = NA creates automatic threshold from data).

Value

Returns a vector of variable numbers corresponding to the model having lowest prediction error.

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

V. Centner, D. Massart, O. de Noord, S. de Jong, B. Vandeginste, C. Sterna, Elimination of uninformative variables for multivariate calibration, Analytical Chemistry 68 (1996) 3851-3858.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(gasoline, package = "pls")
with( gasoline, mcuve_pls(octane, NIR) )
```

`mvrV` *Multivariate regression function*

Description

Adaptation of `mvr` from package `pls` v 2.4.3.

Usage

```
mvrV(
  formula,
  ncomp,
  Y.add,
  data,
  subset,
  na.action,
  shrink,
  method = c("truncation", "stpls", "model.frame"),
  scale = FALSE,
  validation = c("none", "CV", "LOO"),
  model = TRUE,
  x = FALSE,
  y = FALSE,
  ...
)
```

Arguments

`formula`	a model formula. Most of the lm formula constructs are supported. See below.
`ncomp`	the number of components to include in the model (see below).
`Y.add`	a vector or matrix of additional responses containing relevant information about the observations. Only used for cppls.
`data`	an optional data frame with the data to fit the model from.
`subset`	an optional vector specifying a subset of observations to be used in the fitting process.
`na.action`	a function which indicates what should happen when the data contain missing values. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful. See na.omit for other alternatives.
`shrink`	optional shrinkage parameter for `stpls`.
`method`	the multivariate regression method to be used. If "model.frame", the model frame is returned.
`scale`	numeric vector, or logical. If numeric vector, X is scaled by dividing each variable with the corresponding element of scale. If scale is TRUE, X is scaled by dividing each variable by its sample standard deviation. If cross-validation is selected, scaling by the standard deviation is done for every segment.
`validation`	character. What kind of (internal) validation to use. See below.
`model`	a logical. If TRUE, the model frame is returned.
`x`	a logical. If TRUE, the model matrix is returned.
`y`	a logical. If TRUE, the response is returned.
`...`	additional arguments, passed to the underlying fit functions, and mvrCv.

See Also

mvr

myImagePlot *Matrix plotting*

Description

Plot a heatmap with colorbar.

Usage

```
myImagePlot(x, main, ...)
```

Arguments

x	a matrix to be plotted.
main	header text for the plot.
...	additional arguments (not implemented).

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

T. Mehmood, K.H. Liland, L. Snipen, S. Sæbø, A review of variable selection methods in Partial Least Squares Regression, Chemometrics and Intelligent Laboratory Systems 118 (2012) 62-69.

See Also

VIP (SR/sMC/LW/RC), filterPLSR, shaving, stpls, truncation, bve_pls, ga_pls, ipw_pls, mcuve_pls, rep_pls, spa_pls, lda_from_pls, lda_from_pls_cv, setDA.

Examples

```
myImagePlot(matrix(1:12,3,4), 'A header')
```

plsVarSel *Variable selection in Partial Least Squares*

Description

A large collection of variable selection methods for use with Partial Least Squares. These include all methods in Mehmood et al. 2012 and more. All functions treat numeric responses as regression and factor responses as classification. Default classification is PLS + LDA, but `setDA()` can be used to choose PLS + QDA or PLS with response column maximization.

References

T. Mehmood, K.H. Liland, L. Snipen, S. Sæbø, A review of variable selection methods in Partial Least Squares Regression, Chemometrics and Intelligent Laboratory Systems 118 (2012) 62-69. T. Mehmood, S. Sæbø, K.H. Liland, Comparison of variable selection methods in partial least squares regression, Journal of Chemometrics 34 (2020) e3226.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

rep_pls *Regularized elimination procedure in PLS*

Description

A regularized variable elimination procedure for parsimonious variable selection, where also a stepwise elimination is carried out

Usage

```
rep_pls(y, X, ncomp = 5, ratio = 0.75, VIP.threshold = 0.5, N = 3)
```

Arguments

`y`	vector of response values (`numeric` or `factor`).
`X`	numeric predictor `matrix`.
`ncomp`	integer number of components (default = 5).
`ratio`	the proportion of the samples to use for calibration (default = 0.75).
`VIP.threshold`	thresholding to remove non-important variables (default = 0.5).
`N`	number of samples in the selection matrix (default = 3).

Details

A stability based variable selection procedure is adopted, where the samples have been split randomly into a predefined number of training and test sets. For each split, g, the following stepwise procedure is adopted to select the variables. This implementation does not follow the original publication exactly, but it opens for both regression and classification.

Value

Returns a vector of variable numbers corresponding to the model having lowest prediction error.

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

T. Mehmood, H. Martens, S. Sæbø, J. Warringer, L. Snipen, A partial least squares based algorithm for parsimonious variable selection, Algorithms for Molecular Biology 6 (2011).

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(gasoline, package = "pls")
## Not run:
with( gasoline, rep_pls(octane, NIR) )

## End(Not run)
```

`setDA` *Set chosen Discriminant Analysis*

Description

The default methods is LDA, but QDA and column of maximum prediction can be chosen.

Usage

```
setDA(LQ = NULL)
```

Arguments

`LQ`	character argument 'lda', 'qda', 'max' or NULL

Value

Returns the default set method.

See Also

VIP (SR/sMC/LW/RC), filterPLSR, shaving, stpls, truncation, bve_pls, ga_pls, ipw_pls, mcuve_pls, rep_pls, spa_pls, lda_from_pls, lda_from_pls_cv, setDA.

Examples

```
## Not run:
setDA() # Query 'lda', 'qda' or 'max'
setDA('qda') # Set default method to QDA

## End(Not run)
```

---

shaving *Repeated shaving of variables*

Description

One of five filter methods can be chosen for repeated shaving of a certain percentage of the worst performing variables. Performance of the reduced models are stored and viewable through print and plot methods.

Usage

```
shaving(
  y,
  X,
  ncomp = 10,
  method = c("SR", "VIP", "sMC", "LW", "RC"),
  prop = 0.2,
  min.left = 2,
  comp.type = c("CV", "max"),
  validation = c("CV", 1),
  fixed = integer(0),
  newy = NULL,
  newX = NULL,
  segments = 10,
  plsType = "plsr",
  Y.add = NULL,
  ...
)

## S3 method for class 'shaved'
plot(x, y, what = c("error", "spectra"), index = "min", log = "x", ...)
```

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

Arguments

`y`	vector of response values (`numeric` or `factor`).
`X`	numeric predictor `matrix`.
`ncomp`	integer number of components (default = 10).
`method`	filter method, i.e. SR, VIP, sMC, LW or RC given as `character`.
`prop`	proportion of variables to be removed in each iteration (`numeric`).
`min.left`	minimum number of remaining variables.
`comp.type`	use number of components chosen by cross-validation, `"CV"`, or fixed, `"max"`.
`validation`	type of validation for `plsr`. The default is "CV". If more than one set of CV segments is wanted, use a vector of lenth two, e.g. `c("CV",5)`.
`fixed`	vector of indeces for compulsory/fixed variables that should always be included in the modelling.
`newy`	validation response for RMSEP/error computations.
`newX`	validation predictors for RMSEP/error computations.
`segments`	see `mvr` for documentation of segment choices.
`plsType`	Type of PLS model, "plsr" or "cppls".
`Y.add`	Additional response for CPPLS, see `plsType`.
`...`	additional arguments for `plsr` or `cvsegments`.
`x`	object of class `shaved` for plotting or printing.
`what`	plot type. Default = "error". Alternative = "spectra".
`index`	which iteration to plot. Default = "min"; corresponding to minimum RMSEP.
`log`	logarithmic x (default) or y scale.

Details

Variables are first sorted with respect to some importancemeasure, and usually one of the filter measures described above are used. Secondly, a threshold is used to eliminate a subset of the least informative variables. Then a model is fitted again to the remaining variables and performance is measured. The procedure is repeated until maximum model performance is achieved.

Value

Returns a list object of class `shaved` containing the method type, the error, number of components, and number of variables per reduced model. It also contains a list of all sets of reduced variable sets plus the original data.

Author(s)

Kristian Hovde Liland

See Also

VIP (SR/sMC/LW/RC), filterPLSR, shaving, stpls, truncation, bve_pls, ga_pls, ipw_pls, mcuve_pls, rep_pls, spa_pls, lda_from_pls, lda_from_pls_cv, setDA.

Examples

```
data(mayonnaise, package = "pls")
sh <- shaving(mayonnaise$design[,1], pls::msc(mayonnaise$NIR), type = "interleaved")
pars <- par(mfrow = c(2,1), mar = c(4,4,1,1))
plot(sh)
plot(sh, what = "spectra")
par(pars)
print(sh)
```

simulate_classes *Simulate classes*

Description

Simulate multivariate normal data.

Usage

```
simulate_classes(p, n1, n2)

simulate_data(dims, n1 = 150, n2 = 50)
```

Arguments

p	integer number of variables.
n1	integer number of samples in each of two classes in training/calibration data.
n2	integer number of samples in each of two classes in test/validation data.
dims	a 10 element vector of group sizes.

Details

The class simulation is a straigh forward simulation of mulitvariate normal data into two classes for training and test data, respectively. The data simulation uses a strictly structured multivariate normal simulation for with continuous response data.

Value

Returns a list of predictor and response data for training and testing.

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

T. Mehmood, K.H. Liland, L. Snipen, S. Sæbø, A review of variable selection methods in Partial Least Squares Regression, Chemometrics and Intelligent Laboratory Systems 118 (2012) 62-69. T. Mehmood, S. Sæbø, K.H. Liland, Comparison of variable selection methods in partial least squares regression, Journal of Chemometrics 34 (2020) e3226.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
str(simulate_classes(5,4,4))
```

---

spa_pls *Sub-window permutation analysis coupled with PLS (SwPA-PLS)*

Description

SwPA-PLS provides the influence of each variable without considering the influence of the rest of the variables through sub-sampling of samples and variables.

Usage

```
spa_pls(y, X, ncomp = 10, N = 3, ratio = 0.8, Qv = 10, SPA.threshold = 0.05)
```

Arguments

`y`	vector of response values (`numeric` or `factor`).
`X`	numeric predictor `matrix`.
`ncomp`	integer number of components (default = 10).
`N`	number of Monte Carlo simulations (default = 3).
`ratio`	the proportion of the samples to use for calibration (default = 0.8).
`Qv`	integer number of variables to be sampled in each iteration (default = 10).
`SPA.threshold`	thresholding to remove non-important variables (default = 0.05).

Value

Returns a vector of variable numbers corresponding to the model having lowest prediction error.

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

H. Li, M. Zeng, B. Tan, Y. Liang, Q. Xu, D. Cao, Recipe for revealing informative metabolites based on model population analysis, Metabolomics 6 (2010) 353-361. http://code.google.com/p/spa2010/downloads/list.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(gasoline, package = "pls")
with( gasoline, spa_pls(octane, NIR) )
```

`stpls` *Soft-Threshold PLS (ST-PLS)*

Description

A soft-thresholding step in PLS algorithm (ST-PLS) based on ideas from the nearest shrunken centroid method.

Usage

```
stpls(..., method = c("stpls", "model.frame"))
```

Arguments

`...`	arguments for the underlying `stpls.fit` (see Details) and argumetns passed on to `mvrV`).
`method`	choice between the default `stpls` and alternative `model.frame`.

Details

The ST-PLS approach is more or less identical to the Sparse-PLS presented independently by Lè Cao et al. This implementation is an expansion of code from the pls package. Arguments for `stpls.fit` include `ncomp` and `shrink`, where the forme sets then number of components and the latter is the shrinkage parameter indicating how large proportion of the maximum absolute value of the loadings that should be subtracted from the loadings in the nearest shrunken centroid method.

Value

Returns an object of class mvrV, simliar to to mvr object of the pls package.

Author(s)

Solve Sæbø, Tahir Mehmood, Kristian Hovde Liland.

References

S. Sæbø, T. Almøy, J. Aarøe, A.H. Aastveit, ST-PLS: a multi-dimensional nearest shrunken centroid type classifier via pls, Journal of Chemometrics 20 (2007) 54-62.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(yarn, package = "pls")
st <- stpls(density~NIR, ncomp=5, shrink=c(0.1,0.2), validation="CV", data=yarn)
summary(st)
```

`summary.mvrV` *Summary method for stpls and trunc*

Description

Adaptation of `summary.mvr` from the `pls` package v 2.4.3.

Usage

```
## S3 method for class 'mvrV'
summary(
  object,
  what = c("all", "validation", "training"),
  digits = 4,
  print.gap = 2,
  ...
)
```

Arguments

`object`	an mvrV object
`what`	one of "all", "validation" or "training"
`digits`	integer. Minimum number of significant digits in the output. Default is 4.
`print.gap`	Integer. Gap between coloumns of the printed tables.
`...`	Other arguments sent to underlying methods.

See Also

`summary.mvr`

T2_pls *Hotelling's T^2 based variable selection in PLS – T^2-PLS)*

Description

Variable selection based on the T^2 statistic. A side effect of running the selection is printing of tables and production of plots.

Usage

```
T2_pls(ytr, Xtr, yts, Xts, ncomp = 10, alpha = c(0.2, 0.15, 0.1, 0.05, 0.01))
```

Arguments

ytr	Vector of responses for model training.
Xtr	Matrix of predictors for model training.
yts	Vector of responses for model testing.
Xts	Matrix of predictors for model testing.
ncomp	Number of PLS components.
alpha	Hotelling's T^2 significance levels.

Value

Parameters and variables corresponding to variable selections of minimum error and minimum variable set.

References

Tahir Mehmood, Hotelling T^2 based variable selection in partial least squares regression, Chemometrics and Intelligent Laboratory Systems 154 (2016), pp 23-28

Examples

```
data(gasoline, package = "pls")
library(pls)
if(interactive()){
  t2 <- T2_pls(gasoline$octane[1:40], gasoline$NIR[1:40,],
             gasoline$octane[-(1:40)], gasoline$NIR[-(1:40),],
             ncomp = 10, alpha = c(0.2, 0.15, 0.1, 0.05, 0.01))
  matplot(t(gasoline$NIR), type = 'l', col=1, ylab='intensity')
  points(t2$mv[[1]], colMeans(gasoline$NIR)[t2$mv[[1]]], col=2, pch='x')
  points(t2$mv[[2]], colMeans(gasoline$NIR)[t2$mv[[2]]], col=3, pch='o')
}
```

`truncation` *Trunction PLS*

Description

Distribution based truncation for variable selection in subspace methods for multivariate regression.

Usage

```
truncation(..., Y.add, weights, method = "truncation")
```

Arguments

`...`	arguments passed on to `mvrV`).
`Y.add`	optional additional response vector/matrix found in the input data.
`weights`	optional object weighting vector.
`method`	choice (default = `truncation`).

Details

Loading weights are truncated around their median based on confidence intervals for modelling without replicates (Lenth et al.). The arguments passed to `mvrV` include all possible arguments to `cppls` and the following truncation parameters (with defaults) trunc.pow=FALSE, truncation=NULL, trunc.width=NULL, trunc.weight=0, reorth=FALSE, symmetric=FALSE.

The default way of performing truncation involves the following parameter values: truncation="Lenth", trunc.width=0.95, indicating Lenth's confidence intervals (assymmetric), with a confidence of 95 shrinkage instead of a hard threshold. An alternative truncation strategy can be used with: truncation="quantile", in which a quantile line is used for detecting outliers/inliers.

Value

Returns an object of class mvrV, simliar to to mvr object of the pls package.

Author(s)

Kristian Hovde Liland.

References

K.H. Liland, M. Høy, H. Martens, S. Sæbø: Distribution based truncation for variable selection in subspace methods for multivariate regression, Chemometrics and Intelligent Laboratory Systems 122 (2013) 103-111.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(yarn, package = "pls")
tr <- truncation(density ~ NIR, ncomp=5, data=yarn, validation="CV",
 truncation="Lenth", trunc.width=0.95) # Default truncation
summary(tr)
```

`VIP` *Filter methods for variable selection with Partial Least Squares.*

Description

Various filter methods extracting and using information from `mvr` objects to assign importance to all included variables. Available methods are Significance Multivariate Correlation (sMC), Selectivity Ratio (SR), Variable Importance in Projections (VIP), Loading Weights (LW), Regression Coefficients (RC).

Usage

```
VIP(pls.object, opt.comp, p = dim(pls.object$coef)[1])

SR(pls.object, opt.comp, X)

sMC(pls.object, opt.comp, X, alpha_mc = 0.05)

LW(pls.object, opt.comp)

RC(pls.object, opt.comp)

URC(pls.object, opt.comp)

FRC(pls.object, opt.comp)

mRMR(pls.object, nsel, X)
```

Arguments

`pls.object`	`mvr` object from PLS regression.
`opt.comp`	optimal number of components of PLS model.
`p`	number of variables in PLS model.
`X`	data matrix used as predictors in PLS modelling.
`alpha_mc`	quantile significance for automatic selection of variables in `sMC`.
`nsel`	number of variables to select.

Details

From plsVarSel 0.9.10, the VIP method handles multiple responses correctly, as does the LW method. All other filter methods implemented in this package assume a single response and will give its results based on the first response in multi-response cases.

Value

A vector having the same length as the number of variables in the associated PLS model. High values are associated with high importance, explained variance or relevance to the model.

The sMC has an attribute "quantile", which is the associated quantile of the F-distribution, which can be used as a cut-off for significant variables, similar to the cut-off of 1 associated with the VIP.

Author(s)

Tahir Mehmood, Kristian Hovde Liland, Solve Sæbø.

References

T. Mehmood, K.H. Liland, L. Snipen, S. Sæbø, A review of variable selection methods in Partial Least Squares Regression, Chemometrics and Intelligent Laboratory Systems 118 (2012) 62-69. T. Mehmood, S. Sæbø, K.H. Liland, Comparison of variable selection methods in partial least squares regression, Journal of Chemometrics 34 (2020) e3226.

See Also

`VIP` (SR/sMC/LW/RC), `filterPLSR`, `shaving`, `stpls`, `truncation`, `bve_pls`, `ga_pls`, `ipw_pls`, `mcuve_pls`, `rep_pls`, `spa_pls`, `lda_from_pls`, `lda_from_pls_cv`, `setDA`.

Examples

```
data(gasoline, package = "pls")
library(pls)
pls  <- plsr(octane ~ NIR, ncomp = 10, validation = "LOO", data = gasoline)
comp <- which.min(pls$validation$PRESS)
X    <- unclass(gasoline$NIR)
vip <- VIP(pls, comp)
sr  <- SR (pls, comp, X)
smc <- sMC(pls, comp, X)
lw  <- LW (pls, comp)
rc  <- RC (pls, comp)
urc <- URC(pls, comp)
frc <- FRC(pls, comp)
mrm <- mRMR(pls, 401, X)$score
matplot(scale(cbind(vip, sr, smc, lw, rc, urc, frc, mrm)), type = 'l')
```

WVC_pls *Weighted Variable Contribution in PLS (WVC-PLS)*

Description

This implements the PLS-WVC2 component dependent version of WVC from Lin et al., i.e., using Equations 14, 16 and 19. The implementation is used in T. Mehmood, S. Sæbø, K.H. Liland, Comparison of variable selection methods in partial least squares regression, Journal of Chemometrics 34 (2020) e3226. However, there is a mistake in the notation in Mehmood et al. exchanging the denominator of Equation 19 (w'X'Xw) with (w'X'Yw).

Usage

```
WVC_pls(y, X, ncomp, normalize = FALSE, threshold = NULL)
```

Arguments

`y`	Vector of responses.
`X`	Matrix of predictors.
`ncomp`	Number of components.
`normalize`	Divide WVC vectors by maximum value.
`threshold`	Set loading weights smaller than threshold to 0 and recompute component.

Value

loading weights, loadings, regression coefficients, scores and Y-loadings plus the WVC weights.

References

Variable selection in partial least squares with the weighted variable contribution to the first singular value of the covariance matrix, Weilu Lin, Haifeng Hang, Yingping Zhuang, Siliang Zhang, Chemometrics and Intelligent Laboratory Systems 183 (2018) 113–121.

Examples

```
library(pls)
data(mayonnaise, package = "pls")
wvc <- WVC_pls(factor(mayonnaise$oil.type), mayonnaise$NIR, 10)
wvcNT <- WVC_pls(factor(mayonnaise$oil.type), mayonnaise$NIR, 10, TRUE, 0.5)
old.par <- par(mfrow=c(3,1), mar=c(2,4,1,1))
matplot(t(mayonnaise$NIR), type='l', col=1, ylab='intensity')
matplot(wvc$W[,1:3], type='l', ylab='W')
matplot(wvcNT$W[,1:3], type='l', ylab='W, thr.=0.5')
par(old.par)
```

Index