Package ‘sNPLS’

February 20, 2018

Type Package

Title NPLS Regression with L1 Penalization

Version 0.3.31

Author David Hervas

Maintainer David Hervas <ddhervas@yahoo.es>

Depends R (>= 2.10)

Imports car, ggplot2, ks, MASS, Matrix, parallel, pbapply, plotrix, rgl

Description Tools for performing variable selection in three-way data using N-PLS in combination with L1 penalization. The N-PLS model (Rasmus Bro, 1996 <DOI:10.1002/(SICI)1099-128X(199601)10:1%3C47::AID-CEM400%3E3.0.CO;2-C>) is the natural extension of PLS (Partial Least Squares) to N-way structures, and tries to maximize the covariance between X and Y data arrays. The package also adds variable selection through L1 penalization.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-20 17:29:58 UTC

R topics documented:

`bread` *Bread data*

Description

Evaluation of ten bread with respect to eleven attributes by eight judges (Xbread). The outcome is the salt content of each bread (Ybread).

Usage

```
data(bread)
```

Format

An object of class `list` of length 2.

References

Bro, R, Multi-way Analysis in the Food Industry. Models, Algorithms, and Applications. 1998. PhD thesis, University of Amsterdam (NL) & Royal Veterinary and Agricultural University (DK).

`coef.sNPLS` *Coefficients from a sNPLS model*

Description

Extract coefficients from a sNPLS model

Usage

```
## S3 method for class 'sNPLS'
coef(object, as.matrix = FALSE, ...)
```

Arguments

`object`	A sNPLS model fit
`as.matrix`	Should the coefficients be presented as matrix or vector?
`...`	Further arguments passed to `coef`

Value

A matrix (or vector) of coefficients

`cv_fit` *Internal function for* `cv_snpls`

Description

Internal function for `cv_snpls`

Usage

```
cv_fit(xtrain, ytrain, xval, yval, ncomp, keepJ, keepK, ...)
```

Arguments

`xtrain`	A three-way training array
`ytrain`	A response training matrix
`xval`	A three-way test array
`yval`	A response test matrix
`ncomp`	Number of components for the sNPLS model
`keepJ`	Number of variables to keep for each component
`keepK`	Number of 'times' to keep for each component
`...`	Further arguments passed to sNPLS

Value

Returns the CV mean squared error

---

cv_snpls *Cross-validation for a sNPLS model*

Description

Performs cross-validation for a sNPLS model

Usage

```
cv_snpls(X_npls, Y_npls, ncomp = 1:3, keepJ = 1:ncol(X_npls),
  keepK = 1:dim(X_npls)[3], nfold = 10, parallel = TRUE, free_cores = 2,
  ...)
```

Arguments

`X_npls`	A three-way array containing the predictors.
`Y_npls`	A matrix containing the response.
`ncomp`	A vector with the different number of components to test
`keepJ`	A vector with the different number of selected variables to test
`keepK`	A vector with the different number of selected 'times' to test
`nfold`	Number of folds for the cross-validation
`parallel`	Should the computations be performed in parallel?
`free_cores`	If parallel computations are performed how many cores are left unused
`...`	Further arguments passed to sNPLS

Value

A list with the best parameters for the model and the CV error

Examples

```
## Not run:
X_npls<-array(rpois(7500, 10), dim=c(50, 50, 3))

Y_npls<-matrix(2+0.4*X_npls[,5,1]+0.7*X_npls[,10,1]-0.9*X_npls[,15,1]+
0.6*X_npls[,20,1]- 0.5*X_npls[,25,1]+rnorm(50), ncol=1)

cv1<- cv_snpls(X_npls, Y_npls, ncomp=1:2, keepJ = 1:3, keepK = 1:2, parallel = FALSE)

## End(Not run)
```

`fitted.sNPLS` *Fitted method for sNPLS models*

Description

Fitted method for sNPLS models

Usage

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

Arguments

`object`	A sNPLS model fit
`...`	Further arguments passed to `fitted`

Value

Fitted values for the sNPLS model

`plot.cvsNPLS` *Plot cross validation results for sNPLS objects*

Description

Plot function for visualization of cross validation results for sNPLS models

Usage

```
## S3 method for class 'cvsNPLS'
plot(x, facets = TRUE, ...)
```

Arguments

`x`	A cv_sNPLS object
`facets`	Chose between a facet plot or a 3-D scatter plot
`...`	Arguments passed to `car::scatter3d`

Value

A 3D scatter plot with the results of the cross validation

`plot.repeatcv` *Density plot for repat_cv results*

Description

Plots a grid of slices from the 3-D kernel denity estimates of the repeat_cv function

Usage

```
## S3 method for class 'repeatcv'
plot(x, ...)
```

Arguments

`x`	A repeatcv object
`...`	Further arguments passed to plot

Value

A grid of slices from of a 3-D density plot of the results of the repeated cross-validation

`plot.sNPLS` *Plots for sNPLS model fits*

Description

Different plots for sNPLS model fits

Usage

```
## S3 method for class 'sNPLS'
plot(x, type = "T", comps = c(1, 2), ...)
```

Arguments

`x`	A sNPLS model fit
`type`	The type of plot. One of those: "T", "U", "Wj", "Wk", "time" or "variables"
`comps`	A vector of length two with the components to plot
`...`	Options passed to `plot`

Value

A plot of the type specified in the `type` parameter

plot_T *Internal function for* `plot.sNPLS`

Description

Internal function for `plot.sNPLS`

Usage

```
plot_T(x, comps, xlim = c(min(x$T[, comps[1]]) - diff(range(x$T[,
  comps[1]]))/10, max(x$T[, comps[1]]) + diff(range(x$T[, comps[1]]))/10),
  ylim = c(min(x$T[, comps[2]]) - diff(range(x$T[, comps[2]]))/10, max(x$T[,
  comps[2]]) + diff(range(x$T[, comps[2]]))/10), ...)
```

Arguments

`x`	A sNPLS model fit
`comps`	A vector of length two with the components to plot
`xlim`	Limits of the X axis
`ylim`	Limits of the Y axis
`...`	Options passed to `plot`

Value

A plot of the T matrix of a sNPLS model fit

plot_time *Internal function for* `plot.sNPLS`

Description

Internal function for `plot.sNPLS`

Usage

```
plot_time(x, comps, xlab = "Time", ...)
```

Arguments

`x`	A sNPLS model fit
`comps`	A vector with the components to plot
`xlab`	X-axis label
`...`	Options passed to `plot`

Value

A plot of Wk coefficients for each component

`plot_U` *Internal function for* `plot.sNPLS`

Description

Internal function for `plot.sNPLS`

Usage

```
plot_U(x, comps, ylim = c(min(x$U[, comps[2]]) - diff(range(x$U[,
  comps[2]]))/10, max(x$U[, comps[2]]) + diff(range(x$U[, comps[2]]))/10),
  xlim = c(min(x$U[, comps[1]]) - diff(range(x$U[, comps[1]]))/10, max(x$U[,
  comps[1]]) + diff(range(x$U[, comps[1]]))/10), ...)
```

Arguments

`x`	A sNPLS model fit
`comps`	A vector of length two with the components to plot
`ylim`	Limits of the Y axis
`xlim`	Limits of the X axis
`...`	Options passed to `plot`

Value

A plot of the U matrix of a sNPLS model fit

`plot_variables` *Internal function for* `plot.sNPLS`

Description

Internal function for `plot.sNPLS`

Usage

```
plot_variables(x, comps, xlab = "Variables", ...)
```

Arguments

`x`	A sNPLS model fit
`comps`	A vector with the components to plot
`xlab`	X-axis label
`...`	Options passed to `plot`

Value

A plot of Wj coefficients for each component

`plot_Wj` *Internal function for* `plot.sNPLS`

Description

Internal function for `plot.sNPLS`

Usage

```
plot_Wj(x, comps, xlim = c(min(x$Wj[, comps[1]]) - diff(range(x$Wj[,
  comps[1]]))/10, max(x$Wj[, comps[1]]) + diff(range(x$Wj[, comps[1]]))/10),
  ylim = c(min(x$Wj[, comps[2]]) - diff(range(x$Wj[, comps[2]]))/10,
  max(x$Wj[, comps[2]]) + diff(range(x$Wj[, comps[2]]))/10), ...)
```

Arguments

`x`	A sNPLS model fit
`comps`	A vector of length two with the components to plot
`xlim`	Limits of the X axis
`ylim`	Limits of the Y axis
`...`	Options passed to `plot`

Value

A plot of Wj coefficients

plot_Wk *Internal function for* `plot.sNPLS`

Description

Internal function for `plot.sNPLS`

Usage

```
plot_Wk(x, comps, xlim = c(min(x$Wk[, comps[1]]) - diff(range(x$Wk[,
  comps[1]]))/10, max(x$Wk[, comps[1]]) + diff(range(x$Wk[, comps[1]]))/10),
  ylim = c(min(x$Wk[, comps[2]]) - diff(range(x$Wk[, comps[2]]))/10,
  max(x$Wk[, comps[2]]) + diff(range(x$Wk[, comps[2]]))/10), ...)
```

Arguments

`x`	A sNPLS model fit
`comps`	A vector of length two with the components to plot
`xlim`	Limits of the X axis
`ylim`	Limits of the Y axis
`...`	Options passed to `plot`

Value

A plot of the Wk coefficients

predict.sNPLS *Predict for sNPLS models*

Description

Predict function for sNPLS models

Usage

```
## S3 method for class 'sNPLS'
predict(object, newX, rescale = TRUE, ...)
```

Arguments

`object`	A sNPLS model fit
`newX`	A three-way array containing the new data
`rescale`	Should the prediction be rescaled to the original scale?
`...`	Further arguments passed to `predict`

Value

A matrix with the predictions

`repeat_cv` *Repeated cross-validation for sNPLS models*

Description

Performs repeated cross-validatiodn and represents results in a plot

Usage

```
repeat_cv(X_npls, Y_npls, ncomp = 1:3, keepJ = 1:ncol(X_npls),
  keepK = 1:dim(X_npls)[3], nfold = 10, parallel = TRUE, free_cores = 2,
  times = 30, ...)
```

Arguments

`X_npls`	A three-way array containing the predictors.
`Y_npls`	A matrix containing the response.
`ncomp`	A vector with the different number of components to test
`keepJ`	A vector with the different number of selected variables to test
`keepK`	A vector with the different number of selected 'times' to test
`nfold`	Number of folds for the cross-validation
`parallel`	Should the computations be performed in parallel?
`free_cores`	If parallel computations are performed how many cores are left unused
`times`	Number of repetitions of the cross-validation
`...`	Further arguments passed to cv_snpls

Value

A density plot with the results of the cross-validation and an (invisible) `data.frame` with these results

Rmatrix *R-matrix from a sNPLS model fit*

Description

Builds the R-matrix from a sNPLS model fit

Usage

```
Rmatrix(x)
```

Arguments

`x`	A sNPLS model obtained from `sNPLS`

Value

Returns the R-matrix of the model, needed to compute the coefficients

sNPLS *Fit a sNPLS model*

Description

Fits a N-PLS regression model imposing a L1 penalization on `wj` and `wk` matrices

Usage

```
sNPLS(XN, Y, ncomp = 2, conver = 1e-16, max.iteration = 10000,
  keepJ = rep(ncol(XN), ncomp), keepK = rep(rev(dim(XN))[1], ncomp),
  scale.X = TRUE, center.X = TRUE, scale.Y = TRUE, center.Y = TRUE,
  silent = F)
```

Arguments

`XN`	A three-way array containing the predictors.
`Y`	A matrix containing the response.
`ncomp`	Number of components in the projection
`conver`	Convergence criterion
`max.iteration`	Maximum number of iterations
`keepJ`	Number of variables to keep for each component
`keepK`	Number of 'times' to keep for each component
`scale.X`	Perform unit variance scaling on X?

`center.X`	Perform mean centering on X?
`scale.Y`	Perform unit variance scaling on Y?
`center.Y`	Perform mean centering on Y?
`silent`	Show output?

Value

A fitted sNPLS model

References

C. A. Andersson and R. Bro. The N-way Toolbox for MATLAB Chemometrics & Intelligent Laboratory Systems. 52 (1):1-4, 2000.

Shen, H. and Huang, J. Z. (2008). Sparse principal component analysis via regularized low rank matrix approximation. Journal of Multivariate Analysis 99, 1015-1034

Examples

```
X_npls<-array(rpois(7500, 10), dim=c(50, 50, 3))

Y_npls<-matrix(2+0.4*X_npls[,5,1]+0.7*X_npls[,10,1]-0.9*X_npls[,15,1]+
0.6*X_npls[,20,1]- 0.5*X_npls[,25,1]+rnorm(50), ncol=1)

fit<-sNPLS(X_npls, Y_npls, ncomp=3, keepJ = rep(2,3) , keepK = rep(1,3))
```

`summary.sNPLS` *Summary for sNPLS models*

Description

Summary of a sNPLS model fit

Usage

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

Arguments

`object`	A sNPLS object
`...`	Further arguments passed to summary.default

Value

A summary inclunding number of components, squared error and coefficients of the fitted model

`unfold3w` *Unfolding of three-way arrays*

Description

Unfolds a three-way array into a matrix

Usage

```
unfold3w(x)
```

Arguments

`x`	A three-way array

Value

Returns a matrix with dimensions `dim(x)[1] x dim(x)[2]*dim(x([3]))`

Index