Package ‘sNPLS’

October 14, 2022

Type Package

Title NPLS Regression with L1 Penalization

Version 1.0.27

Author David Hervas

Maintainer David Hervas <ddhervas@yahoo.es>

Depends R (>= 2.10)

Imports clickR, future, future.apply, ggplot2, ggrepel, ks, MASS, Matrix, pbapply

Description Tools for performing variable selection in three-way data using N-PLS in combination with L1 penalization, Selectivity Ratio and VIP scores. The N-PLS model (Rasmus Bro, 1996 <DOI:10.1002/(SICI)1099-128X(199601)10:1%3C47::AID-CM400%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, Selectivity Ratio and VIP scores.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-12-16 12:50:02 UTC

R topics documented:

bread ................................................................. 2
goideS .............................................................. 3
cv_fit .............................................................. 3
cv_snpl ............................................................ 4
fitted.sNPLS ....................................................... 5
plot.cvsNPLS ...................................................... 6
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

### Usage

```r
## 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

---

### Usage

```r
cv_fit(
  xtrain, ytrain, xval, yval, ncomp,
  threshold_j = NULL, threshold_k = NULL,
  keepJ = NULL, keepK = NULL,
  method, ...
)
```
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
- `threshold_j`: Threshold value on Wj. Scaled between [0, 1)
- `threshold_k`: Threshold value on Wk. Scaled between [0, 1)
- `keepJ`: Number of variables to keep for each component, ignored if `threshold_j` is provided
- `keepK`: Number of 'times' to keep for each component, ignored if `threshold_k` is provided
- `method`: Select between sNPLS, sNPLS-SR or sNPLS-VIP
- `...`: 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

```r
cv_snpls(
  X_npls,
  Y_npls,
  ncomp = 1:3,
  samples = 20,
  keepJ = NULL,
  keepK = NULL,
  nfold = 10,
  parallel = TRUE,
  method = "sNPLS",
  ...
)
```
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
samples Number of samples for performing random search in continuous thresholding
keepJ A vector with the different number of selected variables to test for discrete thresholding
keepK A vector with the different number of selected 'times' to test for discrete thresholding
nfold Number of folds for the cross-validation
parallel Should the computations be performed in parallel? Set up strategy first with future::plan()
method Select between sNPLS, sNPLS-SR or sNPLS-VIP
... 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)
#Grid search for discrete thresholding
cv1<- cv_snpls(X_npls, Y_npls, ncomp=1:2, keepJ = 1:3, keepK = 1:2, parallel = FALSE)
#Random search for continuous thresholding
cv2<- cv_snpls(X_npls, Y_n pls, ncomp=1:2, samples=20, parallel = FALSE)

## End(Not run)
Arguments

object A sNPLS model fit

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**

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

**Arguments**

x A cv_sNPLS object

Value

A facet plot with the results of the cross validation

---

**plot.repeatcv**

Density plot for repeat_cv results

**Description**

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

**Usage**

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

**Arguments**

x A repeatcv object

Value

Further arguments passed to plot
**plot.sNPLS**

**Value**

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

**Description**

Different plots for sNPLS model fits

**Usage**

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

**Arguments**

- `x` A sNPLS model fit
- `type` The type of plot. One of those: "T", "U", "Wj", "Wk", "time" or "variables"
- `comps` Vector with the components to plot. It can be of length `ncomp` for types "time" and "variables" and of length 2 otherwise.
- `labels` Should rownames be added as labels to the plot?
- `group` Vector with categorical variable defining groups (optional)
- `...` Not used

**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**

```r
plot_T(x, comps, labels, group = NULL)
```
Arguments

\texttt{x} A sNPLS model fit

\texttt{comps} A vector of length two with the components to plot

\texttt{labels} Should rownames be added as labels to the plot?

\texttt{group} Vector with categorical variable defining groups

Value

A plot of the \( T \) matrix of a sNPLS model fit

\begin{verbatim}
plot_time
\end{verbatim}

\textit{Internal function for \texttt{plot.sNPLS}}

Description

Internal function for \texttt{plot.sNPLS}

Usage

\texttt{plot_time(x, comps)}

Arguments

\texttt{x} A sNPLS model fit

\texttt{comps} A vector with the components to plot

Value

A plot of \( W_k \) coefficients for each component

\begin{verbatim}
plot_U
\end{verbatim}

\textit{Internal function for \texttt{plot.sNPLS}}

Description

Internal function for \texttt{plot.sNPLS}

Usage

\texttt{plot_U(x, comps, labels, group = \texttt{NULL})}
**plot_variables**

*Arguments*
- **x**: A sNPLS model fit
- **comps**: A vector of length two with the components to plot
- **labels**: Should rownames be added as labels to the plot?
- **group**: Vector with categorical variable defining groups

*Value*
- A plot of the U matrix of a sNPLS model fit

---

**plot_Wj**

*Description*
- Internal function for plot.sNPLS

*Usage*
- `plot_Wj(x, comps, labels)`

**Arguments**
- **x**: A sNPLS model fit
- **comps**: A vector with the components to plot

*Value*
- A plot of Wj coefficients for each component
Arguments

- `x`: A sNPLS model fit
- `comps`: A vector of length two with the components to plot
- `labels`: Should rownames be added as labels to the 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, labels)`

Arguments

- `x`: A sNPLS model fit
- `comps`: A vector of length two with the components to plot
- `labels`: Should rownames be added as labels to the plot?

Value

A plot of the Wk coefficients

---

`predict.sNPLS`  
*Predict for sNPLS models*

Description

Predict function for sNPLS models

Usage

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

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

Description

Performs repeated cross-validation and represents results in a plot

Usage

```r
repeat_cv(
  X_npls,
  Y_npls,
  ncomp = 1:3,
  samples = 20,
  keepJ = NULL,
  keepK = NULL,
  nfold = 10,
  times = 30,
  parallel = TRUE,
  method = "sNPLS",
  ...
)
```

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
- **samples**: Number of samples for performing random search in continuous thresholding
- **keepJ**: A vector with the different number of selected variables to test in discrete thresholding
- **keepK**: A vector with the different number of selected ‘times’ to test in discrete thresholding
- **nfold**: Number of folds for the cross-validation
times  Number of repetitions of the cross-validation
parallel  Should the computations be performed in parallel? Set up strategy first with
          future::plan()
method  Select between sNPLS, sNPLS-SR or sNPLS-VIP
...  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 sparsity on wj and wk matrices
sNPLS

Usage

sNPLS(
    XN,
    Y,
    ncomp = 2,
    threshold_j = 0.5,
    threshold_k = 0.5,
    keepJ = NULL,
    keepK = NULL,
    scale.X = TRUE,
    center.X = TRUE,
    scale.Y = TRUE,
    center.Y = TRUE,
    conver = 1e-16,
    max.iteration = 10000,
    silent = F,
    method = "sNPLS"
)

Arguments

XN  A three-way array containing the predictors.
Y   A matrix containing the response.
ncomp Number of components in the projection
threshold_j Threshold value on Wj. Scaled between [0, 1)
threshold_k Threshold value on Wk. scaled between [0, 1)
keepJ Number of variables to keep for each component, ignored if threshold_j is pro-

keepK Number of 'times' to keep for each component, ignored if threshold_k is pro-
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?
conver Convergence criterion
max.iteration Maximum number of iterations
silent Show output?
method Select between L1 penalization (sNPLS), variable selection with Selectivity Ra-
tio (sNPLS-SR) or variable selection with VIP (sNPLS-VIP)

Value

A fitted sNPLS model
References


Examples

```R
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)
#Discrete thresholding
fit <- sNPLS(X_npls, Y_npls, ncomp=3, keepJ = rep(2,3) , keepK = rep(1,3))
#Continuous thresholding
fit2 <- sNPLS(X_npls, Y_npls, ncomp=3, threshold_j=0.5, threshold_k=0.5)
#USe sNPLS-SR method
fit3 <- sNPLS(X_npls, Y_npls, ncomp=3, threshold_j=0.5, threshold_k=0.5, method="sNPLS-SR")
```

SR

Compute Selectivity Ratio for a sNPLS model

Description

Estimates Selectivity Ratio for the different components of a sNPLS model fit

Usage

`SR(model)`

Arguments

- `model` A sNPLS model

Value

A list of data.frames, each of them including the computed Selectivity Ratios for each variable
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 including 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

* datasets
  - bread, 2

bread, 2

coeff.sNPLS, 3
cv_fit, 3
cv_snpls, 4

fitted.sNPLS, 5

plot.cvsNPLS, 6
plot.repeatcv, 6
plot.sNPLS, 7
plot_T, 7
plot_time, 8
plot_U, 8
plot_variables, 9
plot_Wj, 9
plot_Wk, 10
predict.sNPLS, 10

repeat_cv, 11
Rmatrix, 12

sNPLS, 12
SR, 14
summary.sNPLS, 15

unfold3w, 15