Package: ddsPLS (via r-universe)

August 27, 2024

Type Package

Title Data-Driven Sparse Partial Least Squares

Version 1.2.1

Author Hadrien Lorenzo

Maintainer Hadrien Lorenzo <hadrien.lorenzo.2015@gmail.com>

Description A sparse Partial Least Squares implementation which uses soft-threshold estimation of the covariance matrices and therein introduces sparsity. Number of components and regularization coefficients are automatically set.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends foreach, R (>= 2.10)

Imports Rcpp (>= 1.0.5), doParallel, shiny, RColorBrewer

LinkingTo Rcpp, RcppEigen

Suggests knitr, rmarkdown, MASS

VignetteBuilder knitr

RoxygenNote 7.3.0

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

RemoteUrl https://github.com/hlorenzo/ddspls

RemoteRef HEAD

RemoteSha 3a57108fa08076dce37b30812d7b85d61f734174

Contents

`bootstrapWrap` *C++ wrapper for bootstrap function*

Description

The wrapper used to start the bootstrap commands. Not to be used by the user.

Usage

```
bootstrapWrap(
  U,
  V,
  X,
  Y,
  lambdas,
  lambda_prev,
  R,
  n_B,
  doBoot = TRUE,
  n,
  p,
  q,
  n_lambdas,
  lambda0.
)
```

Arguments

`U`	matrix, weights X
`V`	matrix, weights Y
`X`	matrix
`Y`	matrix
`lambdas`	vector, the to be tested values for lambda
`lambda_prev`	vector, the previous selected values for lambda
`R`	integer, the desired number of components
`n_B`	integer, the number of bootstrap samples required
`doBoot`	boolean, whether or not perform bootstrap. Used to build the final model (FALSE)
`n`	integer, the number of observations

p	integer, the number of covariates
q	integer, the number of response variables
n_lambdas	integer, the number of to be tested lambdas
lambda0.	the vector of lambda0

Value

A list

`bootstrap_Rcpp` *C++ implementation of the bootstrap operations*

Description

Start the bootstrap operations. Should not be used by user.

Usage

```
bootstrap_Rcpp(
  U,
  V,
  X,
  Y,
  lambdas,
  lambda_prev,
  R,
  n_B,
  doBoot,
  n,
  p,
  q,
  N_lambdas,
  lambda0
)
```

Arguments

U	The weights for X part.
V	The weights for Y part.
X	The matrix of X part.
Y	The matrix of X part.
lambdas	The to be tested values for lambda.
lambda_prev	The previously selected values for lambda.
R	The number of components to build.

n_B	The number of bootstrap samples to generate and analyse.
doBoot	Wheteher do bootstrap operations.
n	The number of observations.
p	The number of variables of X part.
q	The number of variables of Y part.
N_lambdas	The number of to be tested values for lambda.
lambda0	The vector of lambda0

---

ddsPLS *Data-Driven Sparse Partial Least Squares*

Description

The main function of the package. It does both start the ddsPLS algorithm, using bootstrap analysis. Also it estimates automatically the number of components and the regularization coefficients. One regularization parameter per component only is needed to select both in x and in y. Build the optimal model, of the class ddsPLS. Among the different parameters, the lambda is the vector of parameters that are tested by the algorithm along each component for each bootstrap sample. The total number of bootstrap samples is fixed by the parameter n_B, for this parameter, the more the merrier, even if costs more in computation time. This gives access to 3 S3 methods (summary.ddsPLS, plot.ddsPLS and predict.ddsPLS).

Usage

```
ddsPLS(
  X,
  Y,
  criterion = "diffR2Q2",
  doBoot = TRUE,
  LD = FALSE,
  lambdas = NULL,
  n_B = 50,
  n_lambdas = 100,
  lambda_roof = NULL,
  lowQ2 = 0,
  NCORES = 1,
  errorMin = 1e-09,
  verbose = FALSE
)
```

Arguments

X	matrix, the covariate matrix (n,p).
Y	matrix, the response matrix (n,q).
criterion	character, whether diffR2Q2 to be minimized, default, or Q2 to be maximized.

doBoot	logical, whether performing bootstrap operations, default to TRUE. If equal to FALSE, a model with is built on the parameters lambda and the number of components is the length of this vector. In that context, the parameter n_B is ignored. If equal to TRUE, the ddsPLS algorithm, through bootstrap validation, is started using lambda as a grid and n_B as the total number of bootstrap samples to simulate per component.
LD	Boolean, wether or not consider Low-Dimensional dataset.
lambdas	vector, the to be tested values for lambda. Each value for lambda can be interpreted in terms of correlation allowed in the model. More precisely, a covariate 'x[j]' is not selected if its empirical correlation with all the response variables 'y[1..q]' is below lambda. A response variable 'y[k]' is not selected if its empirical correlation with all the covariates 'x[1..p]' is below lambda. Default to seq(0,1,length.out = 30).
n_B	integer, the number of to be simulated bootstrap samples. Default to 50.
n_lambdas	integer, the number of lambda values. Taken into account only if lambdas is NULL. Default to 100.
lambda_roof	limit value to be considered in the optimization.
lowQ2	real, the minimum value of Q^2_B to accept the current lambda value. Default to 0.0.
NCORES	integer, the number of cores used. Default to 1.
errorMin	real, not to be used.
verbose	boolean, whether to print current results. Defaut to FALSE.

Value

A list with different interesting output describing the built model

See Also

summary.ddsPLS, plot.ddsPLS, predict.ddsPLS

Examples

```
# n <- 100 ; d <- 2 ; p <- 20 ; q <- 2
# phi <- matrix(rnorm(n*d),n,d)
# a <- rep(1,p/4) ; b <- rep(1,p/2)
# X <- phi%*%matrix(c(1*a,0*a,0*b,
#                     1*a,3*b,0*a),nrow = d,byrow = TRUE) + matrix(rnorm(n*p),n,p)
# Y <- phi%*%matrix(c(1,0,
#                     0,0),nrow = d,byrow = TRUE) + matrix(rnorm(n*q),n,q)
# model_ddsPLS <- ddsPLS(X,Y,verbose=TRUE)
```

ddsPLS_App *Applet to start ddsPLS*

Description

Applet to start ddsPLS

Usage

```
ddsPLS_App(...)
```

Arguments

...	Same parameters as ddsPLS

Value

Mainly visual objects, also possible to save plots

modelddsPLSCpp_Rcpp *C++ code to build models, internal function*

Description

Build a ddsPLS model once the bootstrap operations has allowed to find a correct lambda.

Usage

```
modelddsPLSCpp_Rcpp(U, V, X, Y, lambdas, R, n, p, q, lambda0)
```

Arguments

U	The weights for X part.
V	The weights for Y part.
X	The matrix of X part.
Y	The matrix of X part.
lambdas	The to be tested values for lambda.
R	The number of components to build.
n	The number of observations.
p	The number of variables of X part.
q	The number of variables of Y part.
lambda0	The vector of regulation parameters.

`plot.ddsPLS` *Function to plot bootstrap performance results of the ddsPLS algorithm*

Description

Function to plot bootstrap performance results of the ddsPLS algorithm

Usage

```
## S3 method for class 'ddsPLS'
plot(
  x,
  type = "criterion",
  digits = 1,
  legend.position = "topright",
  horiz = TRUE,
  biPlot = FALSE,
  las = 0,
  col = NULL,
  cex.names = 1,
  mar = c(5, 4, 4, 2) + 0.1,
  ...
)
```

Arguments

`x`	A ddsPLS object
`type`	The type of graphics. One of "criterion" (default), "total", "prop", "predict", "Q2r", "Q2", "R2r", "R2", "weightX", "weightY", "loadingX" or "loadingY".
`digits`	double. Rounding of the written explained variance.
`legend.position`	character. Where to put the legend.
`horiz`	boolean. Whether to plot horizontally.
`biPlot`	boolean wether or not to plot one component versus the other.
`las`	numeric in (0,1,2,3): the style of axis labels.
`col`	vector. Mainly to modify bars in weight plots.
`cex.names`	double. Size factor for variable names.
`mar`	vector. The margins for the plot.
`...`	Other plotting parameters to affect the plot.

See Also

`ddsPLS`, `predict.ddsPLS`, `summary.ddsPLS`

`predict.ddsPLS` *Function to predict from ddsPLS objects*

Description

Function to predict from ddsPLS objects

Usage

```
## S3 method for class 'ddsPLS'
predict(
  object,
  X_test = NULL,
  toPlot = FALSE,
  doDiagnosis = T,
  legend.position = "topright",
  cex = 1,
  cex.text = 1,
  ...
)
```

Arguments

`object`	A ddsPLS object.
`X_test`	matrix, a test data-set. If is "NULL", the default value, the predicted values for the train test are returned.
`toPlot`	boolean, wether or not to plot the extreme value test plot. Default to 'TRUE'.
`doDiagnosis`	Yes or no to perform diagnosis.
`legend.position`	character. Where to put the legend.
`cex`	float positive. Number indicating the amount by which plotting symbols should be scaled relative to the default.
`cex.text`	float positive. Number indicating the amount by which plotting text elements should be scaled relative to the default.
`...`	Other parameters

See Also

`ddsPLS`, `plot.ddsPLS`, `summary.ddsPLS`

`print.ddsPLS` *Function to sum up bootstrap performance results of the ddsPLS algorithm*

Description

Function to sum up bootstrap performance results of the ddsPLS algorithm

Usage

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

Arguments

`x`	A ddsPLS object.
`...`	Other parameters to be taken into account.

See Also

`ddsPLS`, `plot.ddsPLS`, `predict.ddsPLS`

`summary.ddsPLS` *Function to sum up bootstrap performance results of the ddsPLS algorithm*

Description

Function to sum up bootstrap performance results of the ddsPLS algorithm

Usage

```
## S3 method for class 'ddsPLS'
summary(
  object,
  return = FALSE,
  plotSelection = FALSE,
  las = 1,
  cex.names = 1,
  digits = 2,
  ...
)
```

Arguments

`object`	A ddsPLS object.
`return`	Wether or not to return the printed values, default to FALSE.
`plotSelection`	boolean. Whether plot the selection variables.
`las`	interger. Parameter for angle of variable names.
`cex.names`	real positive. Which factor zomm the variable names.
`digits`	integer indicating the number of decimal places (round) to be used.
`...`	Other parameters to be taken into account.

See Also

`ddsPLS`, `plot.ddsPLS`, `predict.ddsPLS`

Index