# Package ‘fastPLS’

December 11, 2024

**Type** Package

**Title** A Fast Implementation of Partial Least Square

**Version** 0.2

**Date** 2024-12-09

**Description**
An implementation in 'Rcpp' / 'RcppArmadillo' of Partial Least Square algorithms. This package includes other functions to perform the double cross-validation and a fast correlation.

**License** GPL-3

**Depends** R (>= 2.10.0), Matrix

**Imports** Rcpp (>= 0.12.17), methods

**LinkingTo** Rcpp, RcppArmadillo, Matrix

**Suggests** knitr, rmarkdown

**NeedsCompilation** yes

**Author** Stefano Cacciatore [aut, trl, cre]
(<https://orcid.org/0000-0001-7052-7156>),
Dupe Ojo [aut] (<https://orcid.org/0000-0002-5301-8592>),
Leonardo Tenori [aut] (<https://orcid.org/0000-0001-6438-059X>),
Alessia Vignoli [aut] (<https://orcid.org/0000-0003-4038-6596>)

**Maintainer** Stefano Cacciatore <stefano.cacciatore@icgeb.org>

**Repository** CRAN

**Date/Publication** 2024-12-11 15:30:01 UTC

## Contents

fastcor . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
optim.pls.cv . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
pls . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
pls.double.cv . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
predict.fastPLS . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
transformy . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
ViP . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

**Index** **13**

## fastcor *Fast Correlation Analysis*

### Description

This function perform a fast calculation of the Spearman's correlation coefficient.

### Usage

```
fastcor (a, b=NULL, byrow=TRUE, diag=TRUE)
```

### Arguments

| | |
|---|---|
| a | a matrix of training set cases. |
| b | a matrix of training set cases. |
| byrow | if byrow == T rows are correlated (much faster) else columns |
| diag | if diag == T only the diagonal of the cor matrix is returned (much faster). |

### Value

The output matrix of correlation coefficient.

### Author(s)

Stefano Cacciatore, Leonardo Tenori, Dupe Ojo, Alessia Vignoli

### See Also

optim.pls.cv,pls.double.cv

### Examples

```
data(iris)
data=as.matrix(iris[,-5])
fastcor(data)
```

optim.pls.cv *Cross-Validation with PLS-DA.*

## Description

This function performs a 10-fold cross validation on a given data set using Partial Least Squares (PLS) model. To assess the prediction ability of the model, a 10-fold cross-validation is conducted by generating splits with a ratio 1:9 of the data set. This is achieved by removing 10% of samples prior to any step of the statistical analysis, including PLS component selection and scaling. Best number of component for PLS was carried out by means of 10-fold cross-validation on the remaining 90% selecting the best Q2y value. Permutation testing was undertaken to estimate the classification/regression performance of predictors.

## Usage

```
optim.pls.cv (Xdata,
              Ydata,
              ncomp,
              constrain=NULL,
              scaling = c("centering", "autoscaling","none"),
              method = c("plssvd", "simpls"),
              svd.method = c("irlba", "dc"),
              kfold=10)
```

## Arguments

| | |
|---|---|
| `Xdata` | a matrix of independent variables or predictors. |
| `Ydata` | the responses. If Ydata is a numeric vector, a regression analysis will be performed. If Ydata is factor, a classification analysis will be performed. |
| `ncomp` | the number of latent components to be used for classification. |
| `constrain` | a vector of `nrow(data)` elements. Sample sharing a specific identifier or characteristics will be grouped together either in the training set or in the test set of cross-validation. |
| `scaling` | the scaling method to be used. Choices are "`centering`", "`autoscaling`", or "`none`" (by default = "`centering`"). A partial string sufficient to uniquely identify the choice is permitted. |
| `method` | the algorithm to be used to perform the PLS. Choices are "`plssvd`" or "`simpls`" (by default = "`plssvd`"). A partial string sufficient to uniquely identify the choice is permitted. |
| `svd.method` | the SVD method to be used to perform the PLS. Choices are "`irlba`" or "`dc`" (by default = "`irlba`"). A partial string sufficient to uniquely identify the choice is permitted. |
| `kfold` | number of cross-validations loops. |

## Value

The output of the result is a list with the following components:

- `B` the (p x m x length(ncomp)) array containing the regression coefficients. Each row corresponds to a predictor variable and each column to a response variable. The third dimension of the matrix B corresponds to the number of PLS components used to compute the regression coefficients. If ncomp has length 1, B is just a (p x m) matrix.
- `Ypred` the vector containing the predicted values of the response variables obtained by cross-validation.
- `Yfit` the vector containing the fitted values of the response variables.
- `P` the (p x max(ncomp)) matrix containing the X-loadings.
- `Q` the (m x max(ncomp)) matrix containing the Y-loadings.
- `T` the (ntrain x max(ncomp)) matrix containing the X-scores (latent components)
- `R` the (p x max(ncomp)) matrix containing the weights used to construct the latent components.
- `Q2Y` predicting power of model.
- `R2Y` proportion of variance in Y.
- `R2X` vector containg the explained variance of X by each PLS component.
- `txtQ2Y` a summary of the Q2y values.
- `txtR2Y` a summary of the R2y values.

## Author(s)

Dupe Ojo, Alessia Vignoli, Stefano Cacciatore, Leonardo Tenori

## See Also

`pls`,`pls.double.cv`

## Examples

```
data(iris)
data=iris[,-5]
labels=iris[,5]
pp=optim.pls.cv(data,labels,2:4)
pp$optim_comp
```

## pls *Partial Least Squares.*

### Description

Partial Least Squares (PLS) classification and regression for test set from training set.

### Usage

```
pls (Xtrain,
     Ytrain,
     Xtest = NULL,
     Ytest = NULL,
     ncomp=min(5,c(ncol(Xtrain),nrow(Xtrain))),
     scaling = c(”centering”, ”autoscaling”,”none”),
     method = c(”plssvd”, ”simpls”),
     svd.method = c(”irlba”, ”dc”),
     fit = FALSE,
     proj = FALSE,
     perm.test = FALSE,
     times = 100)
```

### Arguments

`Xtrain` a matrix of training set cases.

`Ytrain` a classification vector.

`Xtest` a matrix of test set cases.

`Ytest` a classification vector.

`ncomp` the number of components to consider.

`scaling` the scaling method to be used. Choices are "`centering`", "`autoscaling`", or "`none`" (by default = "`centering`"). A partial string sufficient to uniquely identify the choice is permitted.

`method` the algorithm to be used to perform the PLS. Choices are "`plssvd`" or "`simpls`" (by default = "`plssvd`"). A partial string sufficient to uniquely identify the choice is permitted.

`svd.method` the SVD method to be used to perform the PLS. Choices are "`irlba`" or "`dc`" (by default = "`irlba`"). A partial string sufficient to uniquely identify the choice is permitted.

`fit` a boolean value to perform the fit.

`proj` a boolean value to perform the fit.

`perm.test` a classification vector.

`times` a classification vector.

## Value

A list with the following components:

`B` the (p x m x length(ncomp)) matrix containing the regression coefficients. Each row corresponds to a predictor variable and each column to a response variable. The third dimension of the matrix B corresponds to the number of PLS components used to compute the regression coefficients. If ncomp has length 1, B is just a (p x m) matrix.

`Q` the (m x max(ncomp)) matrix containing the Y-loadings.

`Ttrain` the (ntrain x max(ncomp)) matrix containing the X-scores (latent components)

`R` the (p x max(ncomp)) matrix containing the weights used to construct the latent components.

`mX` mean X.

`vX` variance X.

`mY` mean Y.

`p` matrix for the independent variable X. This indicates how the original data relates to the latent components.

`m` number of predictor variables

`ncomp` number of components used in the PLS model

`Yfit` the prediction values based on the PLS model

`R2Y` proportion of variance in Y

`classification` a boolgean output is given indicating if the response variable is a classification

`lev` level of response variable Y

`Ypred` the (ntest x m x length(ncomp)) containing the predicted values of the response variables for the observations from Xtest. The third dimension of the matrix Ypred corresponds to the number of PLS components used to compute the regression coefficients.

`P` the (p x max(ncomp)) matrix containing the X-loadings.

`Ttest` ...

## Author(s)

Dupe Ojo, Alessia Vignoli, Stefano Cacciatore, Leonardo Tenori

## See Also

`optim.pls.cv`,`pls.double.cv`

## Examples

```
data(iris)
data=iris[,-5]
labels=iris[,5]
ss=sample(150,15)
```

```
ncomponent=3

z=pls(data[-ss,], labels[-ss], data[ss,], ncomp=ncomponent)
```

---

pls.double.cv *Cross-Validation with PLS-DA.*

---

## Description

This function performs a 10-fold cross validation on a given data set using Partial Least Squares (PLS) model. To assess the prediction ability of the model, a 10-fold cross-validation is conducted by generating splits with a ratio 1:9 of the data set, that is by removing 10% of samples prior to any step of the statistical analysis, including PLS component selection and scaling. Best number of component for PLS was carried out by means of 10-fold cross-validation on the remaining 90% selecting the best Q2y value. Permutation testing was undertaken to estimate the classification/regression performance of predictors.

## Usage

```
pls.double.cv (Xdata,
               Ydata,
               ncomp=min(5,c(ncol(Xdata),nrow(Xdata))),
               constrain=1:nrow(Xdata),
               scaling = c("centering", "autoscaling","none"),
               method = c("plssvd", "simpls"),
              svd.method = c("irlba", "dc"),
               perm.test=FALSE,
               times=100,
               runn=10,
               kfold_inner=10,
               kfold_outer=10)
```

## Arguments

| | |
|---|---|
| Xdata | a matrix. |
| Ydata | the responses. If Ydata is a numeric vector, a regression analysis will be performed. If Ydata is factor, a classification analysis will be performed. |
| ncomp | the number of latent components to be used for classification. |
| constrain | a vector of `nrow(data)` elements. Sample with the same identifying constrain will be split in the training set or in the test set of cross-validation together. |
| scaling | the scaling method to be used. Choices are "`centering`", "`autoscaling`", or "`none`" (by default = "`centering`"). A partial string sufficient to uniquely identify the choice is permitted. |

`method` the algorithm to be used to perform the PLS. Choices are "`plssvd`" or "`simpls`" (by default = "`plssvd`"). A partial string sufficient to uniquely identify the choice is permitted.

`svd.method` the SVD method to be used to perform the PLS. Choices are "`irlba`" or "`dc`" (by default = "`irlba`"). A partial string sufficient to uniquely identify the choice is permitted.

`perm.test` a classification vector.

`times` number of cross-validations with permutated samples

`runn` number of cross-validations loops.

`kfold_inner` if perform the optmization of the number of components.

`kfold_outer` if perform the optmization of the number of components.

## Value

A list with the following components:

`B` the (p x m x length(ncomp)) array containing the regression coefficients. Each row corresponds to a predictor variable and each column to a response variable. The third dimension of the matrix B corresponds to the number of PLS components used to compute the regression coefficients. If ncomp has length 1, B is just a (p x m) matrix.

`Ypred` the vector containing the predicted values of the response variables obtained by cross-validation.

`Yfit` the vector containing the fitted values of the response variables.

`P` the (p x max(ncomp)) matrix containing the X-loadings.

`Q` the (m x max(ncomp)) matrix containing the Y-loadings.

`T` the (ntrain x max(ncomp)) matrix containing the X-scores (latent components)

`R` the (p x max(ncomp)) matrix containing the weights used to construct the latent components.

`Q2Y` predictive power of the model.

`R2Y` proportion of variance in Y.

`R2X` vector containg the explained variance of X by each PLS component.

`txtQ2Y` a summary of the Q2y values.

`txtR2Y` a summary of the R2y values.

## Author(s)

Dupe Ojo, Alessia Vignoli, Stefano Cacciatore, Leonardo Tenori

## See Also

`optim.pls.cv`,`pls`

## Examples

```
data(iris)
data=iris[,-5]
labels=iris[,5]
pp=pls.double.cv(data,labels,2:4)
```

## predict.fastPLS *Prediction Partial Least Squares regression.*

### Description

Partial Least Squares (PLS) regression for test set from training set.

### Usage

```
## S3 method for class 'fastPLS'
predict(object, newdata, Ytest=NULL, proj=FALSE, ...)
```

### Arguments

| | |
|---|---|
| object | a matrix of training set cases. |
| newdata | a matrix of predictor variables X for the test set. |
| Ytest | a vector of the response variable Y from Xtest. |
| proj | projection of the test set. |
| ... | further arguments. Currently not used. |

### Value

A list with the following components:

| | |
|---|---|
| Ypred | the (ntest x m x length(ncomp)) containing the predicted values of the response variables for the observations from Xtest. The third dimension of the matrix Ypred corresponds to the number of PLS components used to compute the regression coefficients. |
| Q2Y | predictive power of model |
| Ttest | the (ntrain x max(ncomp)) matrix containing the X-scores (latent components) |

### Author(s)

Dupe Ojo, Alessia Vignoli, Stefano Cacciatore, Leonardo Tenori

### See Also

optim.pls.cv,pls.double.cv

## Examples

```
data(iris)
data=iris[,-5]
labels=iris[,5]
ss=sample(150,15)
ncomponent=3

z=pls(data[-ss,], labels[-ss],  ncomp=ncomponent)
predict(z,data[ss,],FALSE)
```

---

# transformy *Conversion Classification Vector to Matrix*

---

## Description

This function converts a classification vector into a classification matrix.

## Usage

```
transformy(y)
```

## Arguments

| | |
|---|---|
| y | a vector or factor. |

## Details

This function converts a classification vector into a classification matrix. Different groups are compared amongst each other.

## Value

A matrix.

## Author(s)

Dupe Ojo, Alessia Vignoli, Stefano Cacciatore, Leonardo Tenori

## Examples

```
y=c(1,1,1,1,2,2,2,3,3)
print(y)
z=transformy(y)
print(z)
```

## ViP *Variable Importance in the Projection.*

### Description

Variable Importance in the Projection (VIP) is a score that measures how important a variable is in a Partial Least Squares (PLS) model. VIP scores are used to identify which variables are most important in a model and are often used for variable selection.

### Usage

```
ViP (model)
```

### Arguments

`model` a object returning from the pls function.

### Value

A list with the following components:

`B` the (p x m x length(ncomp)) matrix containing the regression coefficients. Each row corresponds to a predictor variable and each column to a response variable. The third dimension of the matrix B corresponds to the number of PLS components used to compute the regression coefficients. If ncomp has length 1, B is just a (p x m) matrix.

`Ypred` the (ntest x m x length(ncomp)) containing the predicted values of the response variables for the observations from Xtest. The third dimension of the matrix Ypred corresponds to the number of PLS components used to compute the regression coefficients.

`P` the (p x max(ncomp)) matrix containing the X-loadings.

`Q` the (m x max(ncomp)) matrix containing the Y-loadings.

`T` the (ntrain x max(ncomp)) matrix containing the X-scores (latent components)

`R` the (p x max(ncomp)) matrix containing the weights used to construct the latent components.

### Author(s)

Dupe Ojo, Alessia Vignoli, Stefano Cacciatore, Leonardo Tenori

### See Also

`optim.pls.cv`,`pls.double.cv`

## Examples

```
data(iris)
data=as.matrix(iris[,-5])
labels=iris[,5]
pp=pls(data,labels,ncomp = 2)
ViP(pp)
```

# Index

∗ **transformation**
    transformy, 10

fastcor, 2

optim.pls.cv, *2*, 3, *6*, *8*, *9*, *11*

pls, *4*, 5, *8*
pls.double.cv, *2*, *4*, *6*, 7, *9*, *11*
predict.fastPLS, 9

transformy, 10

ViP, 11