Package 'spls'

October 14, 2022

Version 2.2-3

Date 2019-05-04

Title Sparse Partial Least Squares (SPLS) Regression and Classification

Author Dongjun Chung <chungdon@stat.wisc.edu>, Hyonho Chun <chun@stat.wisc.edu>, Sunduz Keles <keles@stat.wisc.edu>

Maintainer Valentin Todorov <valentin.todorov@chello.at>

Depends R (>= 2.14)

Imports MASS, nnet, parallel, pls

Description Provides functions for fitting a sparse partial least squares (SPLS) regression and classification (Chun and Keles (2010) <doi:10.1111/j.1467-9868.2009.00723.x>).

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-04 23:10:03 UTC

R topics documented:

`ci.spls` *Calculate bootstrapped confidence intervals of SPLS coefficients*

Description

Calculate bootstrapped confidence intervals of coefficients of the selected predictors and generate confidence interval plots.

Usage

```
ci.spls( object, coverage=0.95, B=1000,
        plot.it=FALSE, plot.fix="y",
        plot.var=NA, K=object$K, fit=object$fit )
```

Arguments

`object`	A fitted SPLS object.
`coverage`	Coverage of confidence intervals. `coverage` should have a number between 0 and 1. Default is 0.95 (95% confidence interval).
`B`	Number of bootstrap iterations. Default is 1000.
`plot.it`	Plot confidence intervals of coefficients?
`plot.fix`	If `plot.fix="y"`, then plot confidence intervals of the predictors for a given response. If `plot.fix="x"`, then plot confidence intervals of a given predictor across all the responses. Relevant only when `plot.it=TRUE`.
`plot.var`	Index vector of responses (if `plot.fix="y"`) or predictors (if `plot.fix="x"`) to be fixed in `plot.fix`. The indices of predictors are defined among the set of the selected predictors. Relevant only when `plot.it=TRUE`.
`K`	Number of hidden components. Default is to use the same `K` as in the original SPLS fit.
`fit`	PLS algorithm for model fitting. Alternatives are `"kernelpls"`, `"widekernelpls"`, `"simpls"`, or `"oscorespls"`. Default is to use the same PLS algorithm as in the original SPLS fit.

Value

Invisibly returns a list with components:

`cibeta`	A list with as many matrix elements as the number of responses. Each matrix element is p by 2, where i-th row of the matrix lists the upper and lower bounds of the bootstrapped confidence interval of the i-th predictor.
`betahat`	Matrix of original coefficients of the SPLS fit.
`lbmat`	Matrix of lower bounds of confidence intervals (for internal use).
`ubmat`	Matrix of upper bounds of confidence intervals (for internal use).

Author(s)

Dongjun Chung, Hyonho Chun, and Sunduz Keles.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

`correct.spls` and `spls`.

Examples

```
data(mice)
# SPLS with eta=0.6 & 1 hidden components
f <- spls( mice$x, mice$y, K=1, eta=0.6 )
# Calculate confidence intervals of coefficients
ci.f <- ci.spls( f, plot.it=TRUE, plot.fix="x", plot.var=20 )
# Bootstrapped confidence intervals
cis <- ci.f$cibeta
cis[[20]]   # equivalent, 'cis$1422478_a_at'
```

---

coefplot.spls *Plot estimated coefficients of the SPLS object*

Description

Plot estimated coefficients of the selected predictors in the SPLS object.

Usage

```
coefplot.spls( object, nwin=c(2,2),
            xvar=c(1:length(object$A)), ylimit=NA )
```

Arguments

`object`	A fitted SPLS object.
`nwin`	Vector of the number of rows and columns in a plotting area. Default is two rows and two columns, i.e., four plots.
`xvar`	Index of variables to be plotted among the set of the selected predictors. Default is to plot the coefficients of all the selected predictors.
`ylimit`	Range of the y axis (the coefficients) in the plot. If `ylimit` is not specified, the y axis of the plot has the range between the minimum and the maximum of all coefficient estimates.

Details

This plot is useful for visualizing coefficient estimates of a variable for different responses. Hence, the function is applicable only with multivariate response SPLS.

Value

NULL.

Author(s)

Dongjun Chung, Hyonho Chun, and Sunduz Keles.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

`ci.spls`, and `correct.spls` and `plot.spls`.

Examples

```
data(yeast)
# SPLS with eta=0.7 & 8 hidden components
f <- spls( yeast$x, yeast$y, K=8, eta=0.7 )
# Draw estimated coefficient plot of the first four variables
# among the selected predictors
coefplot.spls( f, xvar=c(1:4), nwin=c(2,2) )
```

`correct.spls` *Correct the initial SPLS coefficient estimates based on bootstrapped confidence intervals*

Description

Correct initial SPLS coefficient estimates of the selected predictors based on bootstrapped confidence intervals and draw heatmap of original and corrected coefficient estimates.

Usage

```
correct.spls( object, plot.it=TRUE )
```

Arguments

`object`	An object obtained from the function `ci.spls`.
`plot.it`	Draw the heatmap of original coefficient estimates and corrected coefficient estimates?

Details

The set of the selected variables is updated by setting the coefficients with zero-containing confidence intervals to zero.

Value

Invisibly returns a matrix of corrected coefficient estimates.

Author(s)

Dongjun Chung, Hyonho Chun, and Sunduz Keles.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

`ci.spls`.

Examples

```
data(mice)
# SPLS with eta=0.6 & 1 latent components
f <- spls( mice$x, mice$y, K=1, eta=0.6 )
# Calculate confidence intervals of coefficients
ci.f <- ci.spls(f)
# Corrected coefficient estimates
cf <- correct.spls( ci.f )
cf[20,1:5]
```

`cv.sgpls` *Compute and plot the cross-validated error for SGPLS classification*

Description

Draw heatmap of v-fold cross-validated misclassification rates and return optimal eta (thresholding parameter) and K (number of hidden components).

Usage

```
cv.sgpls( x, y, fold=10, K, eta, scale.x=TRUE, plot.it=TRUE,
        br=TRUE, ftype='iden', n.core=8 )
```

Arguments

`x`	Matrix of predictors.
`y`	Vector of class indices.
`fold`	Number of cross-validation folds. Default is 10-folds.
`K`	Number of hidden components.
`eta`	Thresholding parameter. `eta` should be between 0 and 1.
`scale.x`	Scale predictors by dividing each predictor variable by its sample standard deviation?
`plot.it`	Draw the heatmap of cross-validated misclassification rates?
`br`	Apply Firth's bias reduction procedure?
`ftype`	Type of Firth's bias reduction procedure. Alternatives are `"iden"` (the approximated version) or `"hat"` (the original version). Default is `"iden"`.
`n.core`	Number of CPUs to be used when parallel computing is utilized.

Details

Parallel computing can be utilized for faster computation. Users can change the number of CPUs to be used by changing the argument `n.core`.

Value

Invisibly returns a list with components:

`err.mat`	Matrix of cross-validated misclassification rates. Rows correspond to `eta` and columns correspond to number of components (`K`).
`eta.opt`	Optimal `eta`.
`K.opt`	Optimal `K`.

Author(s)

Dongjun Chung and Sunduz Keles.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

See Also

`print.sgpls`, `predict.sgpls`, and `coef.sgpls`.

Examples

```
    data(prostate)
    set.seed(1)

    # misclassification rate plot. eta is searched between 0.1 and 0.9 and
    # number of hidden components is searched between 1 and 5
    ## Not run:
        cv <- cv.sgpls(prostate$x, prostate$y, K = c(1:5), eta = seq(0.1,0.9,0.1),
            scale.x=FALSE, fold=5)

## End(Not run)



    (sgpls(prostate$x, prostate$y, eta=cv$eta.opt, K=cv$K.opt, scale.x=FALSE))
```

`cv.spls` *Compute and plot cross-validated mean squared prediction error for SPLS regression*

Description

Draw heatmap of v-fold cross-validated mean squared prediction error and return optimal eta (thresholding parameter) and K (number of hidden components).

Usage

```
cv.spls( x, y, fold=10, K, eta, kappa=0.5,
        select="pls2", fit="simpls",
        scale.x=TRUE, scale.y=FALSE, plot.it=TRUE )
```

Arguments

x	Matrix of predictors.
y	Vector or matrix of responses.
fold	Number of cross-validation folds. Default is 10-folds.
K	Number of hidden components.
eta	Thresholding parameter. eta should be between 0 and 1.
kappa	Parameter to control the effect of the concavity of the objective function and the closeness of original and surrogate direction vectors. kappa is relevant only when responses are multivariate. kappa should be between 0 and 0.5. Default is 0.5.
select	PLS algorithm for variable selection. Alternatives are "pls2" or "simpls". Default is "pls2".
fit	PLS algorithm for model fitting. Alternatives are "kernelpls", "widekernelpls", "simpls", or "oscorespls". Default is "simpls".
scale.x	Scale predictors by dividing each predictor variable by its sample standard deviation?
scale.y	Scale responses by dividing each response variable by its sample standard deviation?
plot.it	Draw heatmap of cross-validated mean squared prediction error?

Value

Invisibly returns a list with components:

mspemat	Matrix of cross-validated mean squared prediction error. Rows correspond to eta and columns correspond to the number of components (K).
eta.opt	Optimal eta.
K.opt	Optimal K.

Author(s)

Dongjun Chung, Hyonho Chun, and Sunduz Keles.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

print.spls, plot.spls, predict.spls, and coef.spls.

Examples

```
data(yeast)
set.seed(1)

# MSPE plot. eta is searched between 0.1 and 0.9 and
# number of hidden components is searched between 1 and 10

## Not run:
    cv <- cv.spls(yeast$x, yeast$y, K = c(1:10), eta = seq(0.1,0.9,0.1))

    # Optimal eta and K
    cv$eta.opt
    cv$K.opt
    (spls(yeast$x, yeast$y, eta=cv$eta.opt, K=cv$K.opt))

## End(Not run)
```

`cv.splsda` *Compute and plot cross-validated error for SPLSDA classification*

Description

Draw heatmap of v-fold cross-validated misclassification rates and return optimal eta (thresholding parameter) and K (number of hidden components).

Usage

```
cv.splsda( x, y, fold=10, K, eta, kappa=0.5,
        classifier=c('lda','logistic'), scale.x=TRUE, plot.it=TRUE, n.core=8 )
```

Arguments

`x`	Matrix of predictors.
`y`	Vector of class indices.
`fold`	Number of cross-validation folds. Default is 10-folds.
`K`	Number of hidden components.
`eta`	Thresholding parameter. `eta` should be between 0 and 1.
`kappa`	Parameter to control the effect of the concavity of the objective function and the closeness of original and surrogate direction vectors. `kappa` is relevant only for multicategory classification. `kappa` should be between 0 and 0.5. Default is 0.5.
`classifier`	Classifier used in the second step of SPLSDA. Alternatives are `"logistic"` or `"lda"`. Default is `"lda"`.
`scale.x`	Scale predictors by dividing each predictor variable by its sample standard deviation?
`plot.it`	Draw the heatmap of the cross-validated misclassification rates?
`n.core`	Number of CPUs to be used when parallel computing is utilized.

Details

Parallel computing can be utilized for faster computation. Users can change the number of CPUs to be used by changing the argument `n.core`.

Value

Invisibly returns a list with components:

`err.mat`	Matrix of cross-validated misclassification rates. Rows correspond to `eta` and columns correspond to number of components (`K`).
`eta.opt`	Optimal `eta`.
`K.opt`	Optimal `K`.

Author(s)

Dongjun Chung and Sunduz Keles.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

See Also

`print.splsda`, `predict.splsda`, and `coef.splsda`.

Examples

```
data(prostate)
set.seed(1)
# misclassification rate plot. eta is searched between 0.1 and 0.9 and
# number of hidden components is searched between 1 and 5
## Not run:  cv <- cv.splsda( prostate$x, prostate$y, K = c(1:5), eta = seq(0.1,0.9,0.1),
         scale.x=FALSE, fold=5 )
## End(Not run)

(splsda( prostate$x, prostate$y, eta=cv$eta.opt, K=cv$K.opt, scale.x=FALSE ))
```

---

lymphoma *Lymphoma Gene Expression Dataset*

Description

This is the Lymphoma Gene Expression dataset used in Chung and Keles (2010).

Usage

```
data(lymphoma)
```

Format

A list with two components:

x Gene expression data. A matrix with 62 rows and 4026 columns.

y Class index. A vector with 62 elements.

Details

The lymphoma dataset consists of 42 samples of diffuse large B-cell lymphoma (DLBCL), 9 samples of follicular lymphoma (FL), and 11 samples of chronic lymphocytic leukemia (CLL). DBLCL, FL, and CLL classes are coded in 0, 1, and 2, respectively, in y vector. Matrix x is gene expression data and arrays were normalized, imputed, log transformed, and standardized to zero mean and unit variance across genes as described in Dettling (2004) and Dettling and Beuhlmann (2002). See Chung and Keles (2010) for more details.

Source

Alizadeh A, Eisen MB, Davis RE, Ma C, Lossos IS, Rosenwald A, Boldrick JC, Sabet H, Tran T, Yu X, Powell JI, Yang L, Marti GE, Moore T, Hudson J Jr, Lu L, Lewis DB, Tibshirani R, Sherlock G, Chan WC, Greiner TC, Weisenburger DD, Armitage JO, Warnke R, Levy R, Wilson W, Grever MR, Byrd JC, Botstein D, Brown PO, and Staudt LM (2000), "Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling", *Nature*, Vol. 403, pp. 503–511.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

Dettling M (2004), "BagBoosting for tumor classification with gene expression data", *Bioinformatics*, Vol. 20, pp. 3583–3593.

Dettling M and Beuhlmann P (2002), "Supervised clustering of genes", *Genome Biology*, Vol. 3, pp. research0069.1–0069.15.

Examples

```
data(lymphoma)
lymphoma$x[1:5,1:5]
lymphoma$y
```

---

mice *Mice Dataset*

Description

This is the Mice dataset used in Chun and Keles (2010).

Usage

```
data(mice)
```

Format

A list with two components:

x Marker map data. A matrix with 60 rows and 145 columns.

y Gene expression data. A matrix with 60 rows and 83 columns.

Details

The Mice dataset was published by Lan et al. (2006). Matrix `x` is the marker map consisting of 145 microsatellite markers from 19 non-sex mouse chromosomes. Matrix `y` is gene expression measurements of the 83 transcripts from liver tissues of 60 mice. This group of the 83 transcripts is one of the clusters analyzed by Chun and Keles (2010). See Chun and Keles (2010) for more details.

Source

Lan H, Chen M, Flowers JB, Yandell BS, Stapleton DS, Mata CM, Mui E, Flowers MT, Schueler KL, Manly KF, Williams RW, Kendziorski C, and Attie AD (2006), "Combined expression trait correlations and expression quantitative trait locus mapping", *PLoS Genetics*, Vol. 2, e6.

References

Chun H and Keles S (2009), "Expression quantitative trait loci mapping with multivariate sparse partial least squares regression", *Genetics*, Vol. 182, pp. 79–90.

Examples

```
data(mice)
mice$x[1:5,1:5]
mice$y[1:5,1:5]
```

`plot.spls` *Plot the coefficient path of SPLS regression*

Description

Provide the coefficient path plot of SPLS regression as a function of the number of hidden components (K) when eta is fixed.

Usage

```
## S3 method for class 'spls'
plot( x, yvar=c(1:ncol(x$y)), ... )
```

Arguments

`x`	A fitted SPLS object.
`yvar`	Index vector of responses to be plotted.
`...`	Other parameters to be passed through to generic `plot`.

Details

plot.spls provides the coefficient path plot of SPLS fits. The plot shows how estimated coefficients change as a function of the number of hidden components (K), when eta is fixed at the value used by the original SPLS fit.

Value

NULL.

Author(s)

Dongjun Chung, Hyonho Chun, and Sunduz Keles.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

print.spls, predict.spls, and coef.spls.

Examples

```
data(yeast)
# SPLS with eta=0.7 & 8 hidden components
f <- spls( yeast$x, yeast$y, K=8, eta=0.7 )
# Draw coefficient path plots for the first two responses
plot( f, yvar=c(1:2) )
```

---

predict.sgpls *Make predictions or extract coefficients from a fitted SGPLS model*

Description

Make predictions or extract coefficients from a fitted SGPLS object.

Usage

```
## S3 method for class 'sgpls'
predict( object, newx, type = c("fit","coefficient"),
    fit.type = c("class","response"), ... )
## S3 method for class 'sgpls'
coef( object, ... )
```

Arguments

object	A fitted SGPLS object.
newx	If type="fit", then newx should be the predictor matrix of test dataset. If newx is omitted, then prediction of training dataset is returned. If type="coefficient", then newx can be omitted.
type	If type="fit", fitted values are returned. If type="coefficient", coefficient estimates of SGPLS fits are returned.
fit.type	If fit.type="class", fitted classes are returned. If fit.type="response", fitted probabilities are returned. Relevant only when type="fit".
...	Any arguments for predict.sgpls should work for coef.sgpls.

Details

Users can input either only selected variables or all variables for newx.

Value

Matrix of coefficient estimates if type="coefficient". Matrix of predicted responses if type="fit" (responses will be predicted classes if fit.type="class" or predicted probabilities if fit.type="response").

Author(s)

Dongjun Chung and Sunduz Keles.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

See Also

print.sgpls.

Examples

```
data(prostate)
# SGPLS with eta=0.55 & 3 hidden components
f <- sgpls( prostate$x, prostate$y, K=3, eta=0.55, scale.x=FALSE )
# Print out coefficients
coef.f <- coef(f)
coef.f[ coef.f!=0, ]
# Prediction on the training dataset
(pred.f <- predict( f, type="fit" ))
```

predict.spls *Make predictions or extract coefficients from a fitted SPLS model*

Description

Make predictions or extract coefficients from a fitted SPLS object.

Usage

```
## S3 method for class 'spls'
predict( object, newx, type = c("fit","coefficient"), ... )
## S3 method for class 'spls'
coef( object, ... )
```

Arguments

object	A fitted SPLS object.
newx	If type="fit", then newx should be the predictor matrix of test dataset. If newx is omitted, then prediction of training dataset is returned. If type="coefficient", then newx can be omitted.
type	If type="fit", fitted values are returned. If type="coefficient", coefficient estimates of SPLS fits are returned.
...	Any arguments for predict.spls should work for coef.spls.

Details

Users can input either only selected variables or all variables for newx.

Value

Matrix of coefficient estimates if type="coefficient". Matrix of predicted responses if type="fit".

Author(s)

Dongjun Chung, Hyonho Chun, and Sunduz Keles.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

plot.spls and print.spls.

Examples

```
data(yeast)
# SPLS with eta=0.7 & 8 latent components
f <- spls( yeast$x, yeast$y, K=8, eta=0.7 )
# Coefficient estimates of the SPLS fit
coef.f <- coef(f)
coef.f[1:5,]
# Prediction on the training dataset
pred.f <- predict( f, type="fit" )
pred.f[1:5,]
```

`predict.splsda` *Make predictions or extract coefficients from a fitted SPLSDA model*

Description

Make predictions or extract coefficients from a fitted SPLSDA object.

Usage

```
## S3 method for class 'splsda'
predict( object, newx, type = c("fit","coefficient"),
    fit.type = c("class","response"), ... )
## S3 method for class 'splsda'
coef( object, ... )
```

Arguments

`object`	A fitted SPLSDA object.
`newx`	If `type="fit"`, then `newx` should be the predictor matrix of test dataset. If newx is omitted, then prediction of training dataset is returned. If `type="coefficient"`, then `newx` can be omitted.
`type`	If `type="fit"`, fitted values are returned. If `type="coefficient"`, coefficient estimates of SPLSDA fits are returned.
`fit.type`	If `fit.type="class"`, fitted classes are returned. If `fit.type="response"`, fitted probabilities are returned. Relevant only when `type="fit"`.
`...`	Any arguments for `predict.splsda` should work for `coef.splsda`.

Details

Users can input either only selected variables or all variables for `newx`.

Value

Matrix of coefficient estimates if `type="coefficient"`. Matrix of predicted responses if `type="fit"` (responses will be predicted classes if `fit.type="class"` or predicted probabilities if `fit.type="response"`).

Author(s)

Dongjun Chung and Sunduz Keles.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

See Also

print.splsda.

Examples

```
data(prostate)
# SPLSDA with eta=0.8 & 3 hidden components
f <- splsda( prostate$x, prostate$y, K=3, eta=0.8, scale.x=FALSE )
# Print out coefficients
coef.f <- coef(f)
coef.f[ coef.f!=0, ]
# Prediction on the training dataset
(pred.f <- predict( f, type="fit" ))
```

---

print.sgpls *Print function for a SGPLS object*

Description

Print out SGPLS fit, the number and the list of selected predictors.

Usage

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

Arguments

x	A fitted SGPLS object.
...	Additonal arguments for generic print.

Value

NULL.

Author(s)

Dongjun Chung and Sunduz Keles.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

See Also

`predict.sgpls` and `coef.sgpls`.

Examples

```
data(prostate)
# SGPLS with eta=0.55 & 3 hidden components
f <- sgpls( prostate$x, prostate$y, K=3, eta=0.55, scale.x=FALSE )
print(f)
```

`print.spls` *Print function for a SPLS object*

Description

Print out SPLS fit, the number and the list of selected predictors.

Usage

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

Arguments

`x`	A fitted SPLS object.
`...`	Additonal arguments for generic `print`.

Value

NULL.

Author(s)

Dongjun Chung, Hyonho Chun, and Sunduz Keles.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection," *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

`plot.spls`, `predict.spls`, and `coef.spls`.

Examples

```
data(yeast)
# SPLS with eta=0.7 & 8 hidden components
f <- spls( yeast$x, yeast$y, K=8, eta=0.7 )
print(f)
```

`print.splsda` *Print function for a SPLSDA object*

Description

Print out SPLSDA fits, the number and the list of selected predictors.

Usage

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

Arguments

`x`	A fitted SPLSDA object.
`...`	Additonal arguments for generic `print`.

Value

NULL.

Author(s)

Dongjun Chung and Sunduz Keles.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

See Also

`predict.splsda` and `coef.splsda`.

Examples

```
data(prostate)
# SPLSDA with eta=0.8 & 3 hidden components
f <- splsda( prostate$x, prostate$y, K=3, eta=0.8, scale.x=FALSE )
print(f)
```

`prostate` *Prostate Tumor Gene Expression Dataset*

Description

This is the Prostate Tumor Gene Expression dataset used in Chung and Keles (2010).

Usage

```
data(prostate)
```

Format

A list with two components:

x Gene expression data. A matrix with 102 rows and 6033 columns.

y Class index. A vector with 102 elements.

Details

The prostate dataset consists of 52 prostate tumor and 50 normal samples. Normal and tumor classes are coded in 0 and 1, respectively, in `y` vector. Matrix `x` is gene expression data and arrays were normalized, log transformed, and standardized to zero mean and unit variance across genes as described in Dettling (2004) and Dettling and Beuhlmann (2002). See Chung and Keles (2010) for more details.

Source

Singh D, Febbo P, Ross K, Jackson D, Manola J, Ladd C, Tamayo P, Renshaw A, DAmico A, Richie J, Lander E, Loda M, Kantoff P, Golub T, and Sellers W (2002), "Gene expression correlates of clinical prostate cancer behavior", *Cancer Cell*, Vol. 1, pp. 203–209.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

Dettling M (2004), "BagBoosting for tumor classification with gene expression data", *Bioinformatics*, Vol. 20, pp. 3583–3593.

Dettling M and Beuhlmann P (2002), "Supervised clustering of genes", *Genome Biology*, Vol. 3, pp. research0069.1–0069.15.

Examples

```
data(prostate)
prostate$x[1:5,1:5]
prostate$y
```

sgpls *Fit SGPLS classification models*

Description

Fit a SGPLS classification model.

Usage

```
sgpls( x, y, K, eta, scale.x=TRUE,
       eps=1e-5, denom.eps=1e-20, zero.eps=1e-5, maxstep=100,
       br=TRUE, ftype='iden' )
```

Arguments

x	Matrix of predictors.
y	Vector of class indices.
K	Number of hidden components.
eta	Thresholding parameter. eta should be between 0 and 1.
scale.x	Scale predictors by dividing each predictor variable by its sample standard deviation?
eps	An effective zero for change in estimates. Default is 1e-5.
denom.eps	An effective zero for denominators. Default is 1e-20.
zero.eps	An effective zero for success probabilities. Default is 1e-5.
maxstep	Maximum number of Newton-Raphson iterations. Default is 100.
br	Apply Firth's bias reduction procedure?
ftype	Type of Firth's bias reduction procedure. Alternatives are "iden" (the approximated version) or "hat" (the original version). Default is "iden".

Details

The SGPLS method is described in detail in Chung and Keles (2010). SGPLS provides PLS-based classification with variable selection, by incorporating sparse partial least squares (SPLS) proposed in Chun and Keles (2010) into a generalized linear model (GLM) framework. y is assumed to have numerical values, 0, 1, ..., G, where G is the number of classes subtracted by one.

Value

A sgpls object is returned. print, predict, coef methods use this object.

Author(s)

Dongjun Chung and Sunduz Keles.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

`print.sgpls`, `predict.sgpls`, and `coef.sgpls`.

Examples

```
data(prostate)

# SGPLS with eta=0.6 & 3 hidden components
(f <- sgpls(prostate$x, prostate$y, K=3, eta=0.6, scale.x=FALSE))

# Print out coefficients
coef.f <- coef(f)
coef.f[coef.f!=0, ]
```

`spls` *Fit SPLS regression models*

Description

Fit a SPLS regression model.

Usage

```
spls( x, y, K, eta, kappa=0.5, select="pls2", fit="simpls",
    scale.x=TRUE, scale.y=FALSE, eps=1e-4, maxstep=100, trace=FALSE)
```

Arguments

`x`	Matrix of predictors.
`y`	Vector or matrix of responses.
`K`	Number of hidden components.
`eta`	Thresholding parameter. `eta` should be between 0 and 1.
`kappa`	Parameter to control the effect of the concavity of the objective function and the closeness of original and surrogate direction vectors. `kappa` is relevant only when responses are multivariate. `kappa` should be between 0 and 0.5. Default is 0.5.
`select`	PLS algorithm for variable selection. Alternatives are `"pls2"` or `"simpls"`. Default is `"pls2"`.

`fit`	PLS algorithm for model fitting. Alternatives are `"kernelpls"`, `"widekernelpls"`, `"simpls"`, or `"oscorespls"`. Default is `"simpls"`.
`scale.x`	Scale predictors by dividing each predictor variable by its sample standard deviation?
`scale.y`	Scale responses by dividing each response variable by its sample standard deviation?
`eps`	An effective zero. Default is 1e-4.
`maxstep`	Maximum number of iterations when fitting direction vectors. Default is 100.
`trace`	Print out the progress of variable selection?

Details

The SPLS method is described in detail in Chun and Keles (2010). SPLS directly imposes sparsity on the dimension reduction step of PLS in order to achieve accurate prediction and variable selection simultaneously. The option `select` refers to the PLS algorithm for variable selection. The option `fit` refers to the PLS algorithm for model fitting and `spls` utilizes algorithms offered by the **pls** package for this purpose. See help files of the function `plsr` in the **pls** package for more details. The user should install the **pls** package before using **spls** functions. The choices for `select` and `fit` are independent.

Value

A **spls** object is returned. print, plot, predict, coef, ci.spls, coefplot.spls methods use this object.

Author(s)

Dongjun Chung, Hyonho Chun, and Sunduz Keles.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

`print.spls`, `plot.spls`, `predict.spls`, `coef.spls`, `ci.spls`, and `coefplot.spls`.

Examples

```
data(yeast)
# SPLS with eta=0.7 & 8 hidden components
(f <- spls(yeast$x, yeast$y, K=8, eta=0.7))

# Print out coefficients
coef.f <- coef(f)
coef.f[,1]

# Coefficient path plot
plot(f, yvar=1)
```

```
dev.new()

# Coefficient plot of selected variables
coefplot.spls(f, xvar=c(1:4))
```

---

splsda *Fit SPLSDA classification models*

Description

Fit a SPLSDA classification model.

Usage

```
splsda( x, y, K, eta, kappa=0.5,
    classifier=c('lda','logistic'), scale.x=TRUE, ... )
```

Arguments

x	Matrix of predictors.
y	Vector of class indices.
K	Number of hidden components.
eta	Thresholding parameter. eta should be between 0 and 1.
kappa	Parameter to control the effect of the concavity of the objective function and the closeness of original and surrogate direction vectors. kappa is relevant only for multicategory classification. kappa should be between 0 and 0.5. Default is 0.5.
classifier	Classifier used in the second step of SPLSDA. Alternatives are "logistic" or "lda". Default is "lda".
scale.x	Scale predictors by dividing each predictor variable by its sample standard deviation?
...	Other parameters to be passed through to spls.

Details

The SPLSDA method is described in detail in Chung and Keles (2010). SPLSDA provides a two-stage approach for PLS-based classification with variable selection, by directly imposing sparsity on the dimension reduction step of PLS using sparse partial least squares (SPLS) proposed in Chun and Keles (2010). y is assumed to have numerical values, 0, 1, ..., G, where G is the number of classes subtracted by one. The option classifier refers to the classifier used in the second step of SPLSDA and splsda utilizes algorithms offered by **MASS** and **nnet** packages for this purpose. If classifier="logistic", then either logistic regression or multinomial regression is used. Linear discriminant analysis (LDA) is used if classifier="lda". splsda also utilizes algorithms offered by the **pls** package for fitting spls. The user should install **pls**, **MASS** and **nnet** packages before using splsda functions.

Value

A `splsda` object is returned. print, predict, coef methods use this object.

Author(s)

Dongjun Chung and Sunduz Keles.

References

Chung D and Keles S (2010), "Sparse partial least squares classification for high dimensional data", *Statistical Applications in Genetics and Molecular Biology*, Vol. 9, Article 17.

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

See Also

`print.splsda`, `predict.splsda`, and `coef.splsda`.

Examples

```
data(prostate)
# SPLSDA with eta=0.8 & 3 hidden components
f <- splsda( prostate$x, prostate$y, K=3, eta=0.8, scale.x=FALSE )
print(f)
# Print out coefficients
coef.f <- coef(f)
coef.f[ coef.f!=0, ]
```

---

yeast *Yeast Cell Cycle Dataset*

Description

This is the Yeast Cell Cycle dataset used in Chun and Keles (2010).

Usage

```
data(yeast)
```

Format

A list with two components:

x ChIP-chip data. A matrix with 542 rows and 106 columns.

y Cell cycle gene expression data. A matrix with 542 rows and 18 columns.

Details

Matrix y is cell cycle gene expression data (Spellman et al., 1998) of 542 genes from an α factor based experiment. Each column corresponds to mRNA levels measured at every 7 minutes during 119 minutes (a total of 18 measurements). Matrix x is the chromatin immunoprecipitation on chip (ChIP-chip) data of Lee et al. (2002) and it contains the binding information for 106 transcription factors. See Chun and Keles (2010) for more details.

Source

Lee TI, Rinaldi NJ, Robert F, Odom DT, Bar-Joseph Z, Gerber GK, Hannett NM, Harbison CT, Thomson CM, Simon I, Zeitlinger J, Jennings EG, Murray HL, Gordon DB, Ren B, Wyrick JJ, Tagne JB, Volkert TL, Fraenkel E, Gifford DK, and Young RA (2002), "Transcriptional regulatory networks in *Saccharomyces cerevisiae*", *Science*, Vol. 298, pp. 799–804.

Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, Eisen MB, Brown PO, Botstein D, and Futcher B (1998), "Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hydrization", *Molecular Biology of the Cell*, Vol. 9, pp. 3273–3279.

References

Chun H and Keles S (2010), "Sparse partial least squares for simultaneous dimension reduction and variable selection", *Journal of the Royal Statistical Society - Series B*, Vol. 72, pp. 3–25.

Examples

```
data(yeast)
yeast$x[1:5,1:5]
yeast$y[1:5,1:5]
```

Index