Package ‘palasso’

November 19, 2019

Version 0.0.7
Title Paired Lasso Regression
Description Implements sparse regression with paired covariates (Rauschenberger et al. 2019 <doi:10.1007/s11634-019-00375-6>). For the optional shrinkage, install ashr (<https://github.com/stephens999/ashr>) and CorShrink (<https://github.com/kkdey/CorShrink>) from GitHub (see README).

Depends R (>= 3.0.0)
Imports glmnet, Matrix, survival
Suggests knitr, testthat
Enhances ashr, CorShrink, pROC, edgeR
VignetteBuilder knitr
License GPL-3
Encoding UTF-8
Language en-GB
RoxygenNote 7.0.0
LazyData false

URL https://github.com/rauschenberger/palasso

BugReports https://github.com/rauschenberger/palasso/issues

NeedsCompilation no

Author Armin Rauschenberger [aut, cre]
Maintainer Armin Rauschenberger <armin.rauschenberger@uni.lu>

Repository CRAN
Date/Publication 2019-11-19 12:00:02 UTC

R topics documented:

  .args ................................................................. 2
  .combine .................................................................. 3
  .cor ...................................................................... 4
Arguments

Checks the validity of the provided arguments.

Usage

`.args(...)`

Arguments

... Arguments supplied to `palasso`, other than `y`, `X` and `max`.

Value

Returns the arguments as a list, including default values for missing arguments.

Examples

NA
.combine

Combining p-values

Description

This function combines local p-values to a global p-value.

Usage

\texttt{.combine(x, method = "simes")}

Arguments

- \texttt{x} \quad \text{local p-values: numeric vector of length} k
- \texttt{method} \quad \text{character } "\text{fisher}" , "\text{tippet}" , "\text{sidak}" , \text{or } "\text{simes}"

Value

These functions return a numeric vector of length \( p \) (main effects), or a numeric matrix with \( p \) rows and \( p \) columns (interaction effects).

References


Examples

\begin{verbatim}
# independence
p <- runif(10)
palasso:::.combine(p)

## dependence
#runif <- function(n,cor=0){
#  Sigma <- matrix(cor,nrow=n,ncol=n)
#  diag(Sigma) <- 1
#  mu <- rep(0,times=n)
#  q <- MASS::mvrnorm(n=1,mu=mu,Sigma=Sigma)
#  stats::pnorm(q=q)
#}
#p <- runif(n=10,cor=0.8)
#combine(p)
\end{verbatim}
.cor  
Correlation

Description
Calculates the correlation between the response and the covariates. Shrinks the correlation coefficients for each covariate set separately.

Usage
.cor(y, x, args)

Arguments
y vector of length n
x matrix with n rows and p columns
args options for paired lasso: list of arguments (output from .dims and .args)

Value
list of vectors

Examples
NA

.cv  
Cross-validation

Description
Repeatedly leaves out samples, and predicts their response.

Usage
.cv(y, x, foldid, lambda, args)

Arguments
y response: vector of length n
x covariates: matrix with n rows (samples) and k * p columns (variables)
foldid fold identifiers: vector of length n, with entries from 1 to nfolds
lambda lambda sequence: vector of decreasing positive values
args options for paired lasso: list of arguments (output from .dims and .args)
`.dims`  

**Value**  

Returns matrix of predicted values (except "cox")

**Examples**  

NA

---

<table>
<thead>
<tr>
<th><code>.dims</code></th>
<th><strong>Dimensionality</strong></th>
</tr>
</thead>
</table>

**Description**  

This function extracts the dimensions.

**Usage**  

```
.dims(y, X, args = NULL)
```

**Arguments**  

- `y`  
  response: vector of length `n`
- `X`  
  covariates: list of matrices, each with `n` rows (samples) and `p` columns (variables)
- `args`  
  options for paired lasso: list of arguments (output from `.dims` and `.args`)

**Value**  

The function `.dims` extracts the dimensionality. It returns the numbers of samples, covariate pairs and covariate sets. It also returns the number of weighting schemes, and the names of these weighting schemes.

**Examples**  

NA
.extract

Description

Extracts cv.glmnet-like object.

Usage

.extract(fit, lambda, cvm, type.measure)

Arguments

- **fit**: matrix with one row for each sample ("gaussian", "binomial" and "poisson"), or one row for each fold (only "cox"), and one column for each lambda (output from .fit)
- **lambda**: lambda sequence: vector of decreasing positive values
- **cvm**: mean cross-validated loss: vector of same length as lambda (output from .loss)
- **type.measure**: ... loss function: character "deviance", "mse", "mae", "class", or "auc"

Examples

NA

.fit

Description

Fits all models from the chosen bag.

Usage

.fit(y, x, args)

Arguments

- **y**: response: vector of length $n$
- **x**: covariates: matrix with $n$ rows (samples) and $k \times p$ columns (variables)
- **args**: options for paired lasso: list of arguments (output from .dims and .args)

Value

list of glmnet-like objects
### .folds

#### Description

Assigns samples to cross-validation folds, balancing the folds in the case of a binary or survival response.

#### Usage

\[
\text{.folds}(y, \text{nfolds}, \text{foldid} = \text{NULL})
\]

#### Arguments

- **y**: response; vector of length \(n\)
- **nfolds**: number of folds; positive integer (\(\geq 10\) recommended)
- **foldid**: fold identifiers; vector of length \(n\), with entries from 1 to \(\text{nfolds}\)

#### Value

Returns the fold identifiers.

#### Examples

NA

### .loss

#### Description

Calculates mean cross-validated loss

#### Usage

\[
\text{.loss}(y, \text{fit}, \text{family}, \text{type.measure}, \text{foldid} = \text{NULL})
\]
Arguments

y response: vector of length \( n \)

fit matrix with one row for each sample ("gaussian", "binomial" and "poisson"), or one row for each fold (only "cox"), and one column for each \( \lambda \) (output from \texttt{.fit})

family model family: character "gaussian", "binomial", "poisson", or "cox"

type.measure ... loss function: character "deviance", "mse", "mae", "class", or "auc"

foldid fold identifiers: vector of length \( n \), with entries from 1 to \( n_{folds} \)

Value

Returns list of vectors, one for each model.

Examples

NA

---

\texttt{.weight} \hspace{1cm} \textit{Weighting schemes}

Description

Calculates the weighting schemes.

Usage

\texttt{.weight(cor, args)}

Arguments

cor correlation coefficients: list of \( k \) vectors of length \( p \) (one vector for each covariate set with one entry for each covariate)

args options for paired lasso: list of arguments (output from \texttt{.dims} and \texttt{.args})

Value

list of named vectors (one for each weighting scheme)

Examples

NA
## Arguments

This page lists the arguments for the (internal) "palasso" function(s).

### Arguments

- **y**
  
  response: vector of length \( n \)

- **X**
  
  covariates: list of matrices, each with \( n \) rows (samples) and \( p \) columns (variables)

- **max**
  
  maximum number of non-zero coefficients: positive numeric, or NULL (no sparsity constraint)

- **...**
  
  further arguments for `cv.glmnet` or `glmnet`

- **x**
  
  covariates: matrix with \( n \) rows (samples) and \( k \times p \) columns (variables)

- **args**
  
  options for paired lasso: list of arguments (output from `.dims` and `.args`)

- **nfolds**
  
  number of folds: positive integer (\( \geq 10 \) recommended)

- **foldid**
  
  fold identifiers: vector of length \( n \), with entries from 1 to \( nfolds \)

- **cor**
  
  correlation coefficients: list of \( k \) vectors of length \( p \) (one vector for each covariate set with one entry for each covariate)

- **lambda**
  
  lambda sequence: vector of decreasing positive values

- **family**
  
  model family: character "gaussian", "binomial", "poisson", or "cox"

- **type.measure**
  
  ... loss function: character "deviance", "mse", "mae", "class", or "auc"

- **fit**
  
  matrix with one row for each sample ("gaussian", "binomial" and "poisson"), or one row for each fold (only "cox"), and one column for each lambda (output from `.fit`)

- **cvm**
  
  mean cross-validated loss: vector of same length as lambda (output from `.loss`)

## Methods

This page lists the main methods for class "palasso".
Usage

```r
## S3 method for class 'palasso'
predict(object, newdata, model = "paired", s = "lambda.min", max = NULL, ...)

## S3 method for class 'palasso'
coef(object, model = "paired", s = "lambda.min", max = NULL, ...)

## S3 method for class 'palasso'
weights(object, model = "paired", max = NULL, ...)

## S3 method for class 'palasso'
fitted(object, model = "paired", s = "lambda.min", max = NULL, ...)

## S3 method for class 'palasso'
residuals(object, model = "paired", s = "lambda.min", max = NULL, ...)

## S3 method for class 'palasso'
deviance(object, model = "paired", max = NULL, ...)

## S3 method for class 'palasso'
logLik(object, model = "paired", max = NULL, ...)

## S3 method for class 'palasso'
summary(object, model = "paired", ...)
```

Arguments

- `object` palasso object
- `newdata` covariates: list of matrices, each with `n` rows (samples) and `p` columns (variables)
- `model` character "paired", or an entry of `names(object)`
- `s` penalty parameter: character "lambda.min" or "lambda.1se", positive numeric, or NULL (entire sequence)
- `max` maximum number of non-zero coefficients, positive integer, or NULL
- `...` further arguments for `predict.cv.glmnet`, `coef.cv.glmnet`, or `deviance.glmnet`

Details

By default, the function `predict` returns the linear predictor (type="link"). Consider predicting the response (type="response").

See Also

Use `palasso` to fit the paired lasso.
**Description**

The function `palasso` fits the paired lasso. Use this function if you have paired covariates and want a sparse model.

**Usage**

`palasso(y = y, X = X, max = 10, ...)`

**Arguments**

- `y`: response; vector of length `n`
- `X`: covariates; list of matrices, each with `n` rows (samples) and `p` columns (variables)
- `max`: maximum number of non-zero coefficients; positive numeric, or NULL (no sparsity constraint)
- `...`: further arguments for `cv.glmnet` or `glmnet`

**Details**

Let `x` denote one entry of the list `X`. See `glmnet` for alternative specifications of `y` and `x`. Among the further arguments, `family` must equal "gaussian", "binomial", "poisson", or "cox", and `penalty.factor` must not be used.

Hidden arguments: Deactivate adaptive lasso by setting `adaptive` to `FALSE`, activate standard lasso by setting `standard` to `TRUE`, and deactivate shrinkage by setting `shrink` to `FALSE`.

**Value**

This function returns an object of class `palasso`. Available methods include `predict`, `coef`, `weights`, `fitted`, `residuals`, `deviance`, `logLik`, and `summary`.

**References**


**Examples**

```r
set.seed(1)
n <- 50; p <- 20
y <- rbinom(n=n,size=1,prob=0.5)
X <- lapply(1:2,function(x) matrix(rnorm(n=p),nrow=n,ncol=p))
object <- palasso(y=y,X=X,family="binomial",standard=TRUE)
```
Index

*Topic methods
  .combine, 3
  .args, 2, 4–6, 8, 9
  .combine, 3
  .cor, 4
  .cv, 4
  .dims, 4, 5, 5, 6, 8, 9
  .extract, 6
  .fit, 6, 6, 8, 9
  .folds, 7
  .loss, 6, 7, 9
  .weight, 8

arguments, 9

coef, 11
coeff, cv.glmnet, 10
coeff.palasso (methods), 9
cv.glmnet, 9, 11

deviance, 11
deviance.glmnet, 10
deviance.palasso (methods), 9

fitted, 11
fitted.palasso (methods), 9

glmnet, 9, 11

logLik, 11
logLik.palasso (methods), 9

methods, 9

palasso, 2, 10, 11
palasso-package (palasso), 11
predict, 11
predict.cv.glmnet, 10
predict.palasso (methods), 9

residuals, 11
residuals.palasso (methods), 9

summary, 11
summary.palasso (methods), 9

weights, 11
weights.palasso (methods), 9