Package ‘ExactPath’

February 19, 2015

Type Package

Title Exact solution paths for regularized LASSO regressions with L_1 penalty

Version 1.0

Date 2013-02-05

Author Dr. Kai Wang

Maintainer Kai Wang <kai-wang@uiowa.edu>

Depends R(>= 2.12), ncvreg, lars

Description ExactPath implements an algorithm for exact LASSO solution. Two methods are provided to print and visualize the whole solution paths. Use ?ExactPath to see an introduction. Packages ncvreg and lars are required so that their data sets can be used in examples.

License GPL (>= 2)

LazyLoad yes

Repository CRAN

Date/Publication 2013-02-06 08:27:04

NeedsCompilation no

R topics documented:

ExactPath-package .......................................................... 2
exact.path ............................................................... 3
LASSO.exact ............................................................. 4
plot.path ............................................................... 5
print.path ............................................................ 6

Index 8
ExactPath-package

Exact Solution Paths for Regularized L_1 LASSO Regression

Description

ExactPath implements an algorithm for exact LASSO solution. Two methods are provided to print and visualize the whole solution paths. Use ?ExactPath to see an introduction. Packages ncvreg and lars are required so that their data sets can be used in examples.

Details

Package: ExactPath
Type: Package
Version: 1.0
Date: 2013-02-05
License: GPL (>=2)
LazyLoad: yes

Author(s)

Kai Wang <kai-wang@uiowa.edu>

References


Examples

```r
library(ncvreg)
data(prostate)
myfit = exact.path(as.matrix(prostate[, -9]), prostate$lpsa, verbose=TRUE)
myfit
plot(myfit)

library(ncvreg)
data(heart)
myfit = exact.path(as.matrix(heart[, -1]), heart$sbp)
myfit
plot(myfit)

library(lars)
data(diabetes)
myfit = exact.path(diabetes$x, diabetes$y, verbose=TRUE)
myfit
plot(myfit)
```
exact.path

Generate Exact Solution Path

Description

exact.path generates the whole solution paths.

Usage

exact.path(X, y, max.var=20, verbose=FALSE)

Arguments

X    an $n \times p$ design matrix for the predictors.
y    a vector of the response values.
max.var a numerical value (default value: 20) that gives the maximum number of steps. Extra 4 steps are allowed in case the same set of predictors enter or exit the set of active predictors more than once. This value contributes to the determination of how many $\lambda_k$s are to be found.
verbose If TRUE, the value of $\lambda_k$ and the associated information will be printed at every step a new $\lambda_k$ is found. The default is FALSE.

Details

Starting with a large value, this function finds recursively $\lambda_1, \lambda_2, \ldots$ until the desired number of steps is achieved. At each step, inactive predictors become active, active predictors become inactive, or both. The selection indicator is automatically determined. The backend engine is LASSO.exact.

It is not necessary to standardize the columns of $X$ and the response vector $y$. Such standardization is conducted anyway in this function.

Value

A list object of class "path". This list contains the following components:
breaks a length $K$ vector of $\lambda_k$s
tau a $p \times K$ matrix of selection indicators.
beta a $p \times K$ matrix of regression coefficients. See also LASSO.exact.
score a $p \times K$ matrix of scores. See also LASSO.exact.

Author(s)

Kai Wang <kai-wang@uiowa.edu>

References

See Also

LASSO.exact is the function for generating $\lambda_k$ at each step.

Examples

library(ncvreg)
data(prostate)
exact.path(as.matrix(prostate[,9]), prostate$lpsa, verbose=TRUE)

library(ncvreg)
data(heart)
exact.path(as.matrix(heart[,1]), heart$sbp)

library(lars)
data(diabetes)
exact.path(diabetes$x, diabetes$y, verbose=TRUE)

Description

LASSO.exact() generates the next value of $\lambda_k$ and the associated selection indicator vector.

Usage

LASSO.exact(lambda, tau, X, y)

Arguments

lambda  
a numerical value for lambda. It does not need to be one of the $\lambda_k$s.

tau  
a vector of length $p$. Its elements are equal to 1, $-1$, or 0. This is the selection
indicator vector.

X  
an $n \times p$ design matrix for the predictors.

y  
a vector of the response values.

Details

This function computes the next $\lambda_k$ that $\lambda_k < \lambda$. Although it is a stand-alone function, it is meant to be called by exact.path(). exact.path() automatically finds recursively the $\lambda_k$s and sets up the correct values for tau for each $\lambda_k$.

It is not necessary to standardize the columns of $X$ and the response vector $y$. Such standardization is conducted anyway in this function.
Value

Given the value $\lambda \in [\lambda_k, \lambda_{k-1})$ and $\tau(\lambda)$, this function returns a data frame with the following variables:

**beta**
a vector of length $p$. This is the vector of regression coefficients of the predictors evaluated at $\lambda_k$. Note that for active predictors, their coefficients are non-zero; for inactive predictors their coefficients are equal to 0.

**S**
a vector of length $p$. This is the vector of scores of the predictors evaluated at $\lambda_k$. Note that for active predictors, the absolute values of their scores should be equal to $\lambda$; for inactive predictors the absolute value of their scores are $\leq \lambda$.

**breaks**
a vector of length $p$. This is the vector of candidate $\lambda$ values provided by each predictor. For active predictors, this is the value at which its regression coefficient is equal to 0. For inactive predictors, this is the value at which the absolute value of its score is equal to $\lambda$. The largest value of this vector that is less than $\lambda$ is chosen to be $\lambda_k$.

**tau**
a vector of length $p$. This is the vector of selection indicators predicted for $\lambda_k$.

**change**
a vector of length $p$. This is the vector of predicted status changes at $\lambda_k$: a predictor becomes active ("+"), inactive ("-"), or remains unchanged ("|").

Author(s)

Kai Wang <kai-wang@uiowa.edu>

References


See Also

*exact.path* automatically generates the whole solution paths.

Examples

```r
library(lars)
data(diabetes)
LASSO.exact(0.05, c(0,-1,1,1,0,0,-1,0,1,1), diabetes$x, diabetes$y)
```

---

**plot.path**

*Plot Exact Solution Paths from a "path" Object*

Description

plot.path produces the solution paths generated by exact.path.
Usage

### S3 method for class "path"
### S3 method for class 'path'

plot(x, ...)

Arguments

- **x**: an object of class `path`. See also `exact.path`.
- **...**: not used.

Author(s)

Kai Wang <kai-wang@uiowa.edu>

See Also

- `exact.path`

Examples

```r
library(ncvreg)
data(prostate)
plot(exact.path(as.matrix(prostate[,-9]), prostate$lpsa, verbose=TRUE))

library(ncvreg)
data(heart)
plot(exact.path(as.matrix(heart[,-1]), heart$sbp))

library(lars)
data(diabetes)
plot(exact.path(diabetes$x, diabetes$y, verbose=TRUE))
```

print.path

---

**Description**

print.path prints out the content of a "path" object generated by exact.path.

Usage

### S3 method for class 'path'

print(x, ...)

Arguments

- **x**: an object of class `path`. See also `exact.path`.
- **...**: not used.
**print.path**

**Author(s)**
Kai Wang <kai-wang@uiowa.edu>

**See Also**
`exact.path`.

**Examples**

```r
library(ncvreg)
data(prostate)
exact.path(as.matrix(prostate[, -9]), prostate$lpsa, verbose=TRUE)

library(ncvreg)
data(heart)
exact.path(as.matrix(heart[, -1]), heart$sbp)

library(lars)
data(diabetes)
exact.path(diabetes$x, diabetes$y, verbose=TRUE)
```
Index

exact.path, 3, 5–7
ExactPath (ExactPath-package), 2
ExactPath-package, 2

lars, 2
LASSO.exact, 4, 4

ncvreg, 2

plot.path, 5
print.path, 6