Package: scout (via r-universe)

September 8, 2024

Type Package

Title Implements the Scout Method for Covariance-Regularized Regression

Version 1.0.4

Date 2015-07-09

Author Daniela M. Witten and Robert Tibshirani

Maintainer Daniela M. Witten <dwitten@uw.edu>

Imports glasso, stats, grDevices, graphics

Suggests lars

Description Implements the Scout method for regression, described in ``Covariance-regularized regression and classification for high-dimensional problems'', by Witten and Tibshirani (2008), Journal of the Royal Statistical Society, Series B 71(3): 615-636.

License GPL (>= 2)

NeedsCompilation yes

Date/Publication 2015-07-10 20:41:38

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

RemoteUrl https://github.com/cran/scout

RemoteRef HEAD

RemoteSha f4bc2ffc7efa1bfa1e95d88ed703c5a19ef82c89

Contents

`scout-package` *Implements covariance-regularized regression, aka the Scout Method.*

Description

Functions for implementing covariance-regularize regression.

Details

Package:	scout
Type:	Package
Version:	1.0
Date:	2008-11-20
License:	GPL (>=2)

The main function is "scout", which takes in a data matrix x and an outcome vector y and estimates regression coefficients for Scout(2,1) for a range of tuning parameter values. Alternatively one can specify other tuning parameter values and one can also perform Scout(1,1), Scout(2,.), or Scout(1,.). Cross-validation and prediction functions also are available.

Author(s)

Daniela Witten and Robert Tibshirani

Maintainer: Daniela Witten <dwitten@stanford.edu>

References

Witten and Tibshirani (2008) Covariance-regularized regression and classification for high-dimensional problems. Journal of the Royal Statistical Society, Series B 71(3): 615-636.

See Also

<http://www-stat.stanford.edu/~dwitten>

Examples

```
library(lars)
data(diabetes)
attach(diabetes)
## Not run: cv.out <- cv.scout(x2,y,p1=1,p2=1,K=3)
## Not run: print(cv.out)
## Not run: out <- scout(x2,y,p1=1,p2=1,lam1=cv.out$bestlam1,lam2=cv.out$bestlam2)
## Not run: coef <- out$coef[1,1,]
detach(diabetes)
```

crossProdLasso *Performs the lasso on the cross product matrices X'X and X'y*

Description

Perform L1-regularized regression of y onto X using only the cross-product matrices X'X and X'y. In the case of covariance-regularized regression, this is useful if you would like to try out something other than L1 or L2 regularization of the inverse covariance matrix.

Suppose you use your own method to regularize X'X. Then let Sigma denote your estimate of the population covariance matrix. Now say you want to minimize beta' Sigma beta - 2 beta' X'y + lambda ||beta||_1 in order to get the regression estimate beta, which maximizes the second scout criterion when an L_1 penalty is used. You can do this by calling crossProdLasso(Sigma, X'y,rho).

If you run crossProdLasso(X'X,X'y,rho) then it should give the same result as lars(X,y)

Notice that the xtx that you pass into this function must be POSITIVE SEMI DEFINITE (or positive definite) or the problem is not convex and the algorithm will not converge.

Usage

```
crossProdLasso(xtx,xty,rho,thr=1e-4,maxit=100,beta.init=NULL)
```

Arguments

`xtx`	A pxp matrix, which should be an estimate of a covariance matrix. This matrix must be POSITIVE SEMI DEFINITE (or positive definite) or the problem is not convex and the algorithm will not converge.
`xty`	A px1 vector, which is generally obtained via X'y.
`rho`	Must be non-negative; the regularization parameter you are using.
`thr`	Convergence threshold.
`maxit`	How many iterations to perform?
`beta.init`	If you're running this over a range of rho values, then set beta.init equal to the solution you got for a previous rho value. It will speed things up.

Details

If your xtx is simply X'X for some X, and your xty is simple X'y with some y, then the results will be the same as running lars on data (X,y) for a single shrinkage parameter value.

Note that when you use the scout function with p2=1, the crossProdLasso function is called internally to give the regression coefficients, after the regularized inverse covariance matrix is estimated. It is provided here in case it is useful to the user in other settings.

Value

`beta`	A px1 vector with the regression coefficients.

Note

The FORTRAN code that this function links to was kindly written and provided by Jerry Friedman.

Author(s)

FORTRAN code by Jerry Friedman. R interface by Daniela M. Witten and Robert Tibshirani

References

Witten, DM and Tibshirani, R (2008) Covariance-regularized regression and classification for high-dimensional problems. Journal of the Royal Statistical Society, Series B 71(3): 615-636. <http://www-stat.stanford.edu/~dwitten>

Examples

```
set.seed(1)
#data(diabetes)
#attach(diabetes)
x2 <- matrix(rnorm(10*20),ncol=20)
y <- rnorm(10)
# First, let's do scout(2,1) the usual way).
scout.out <- scout(x2,y,p1=2,p2=1)
print(scout.out)



# Now, suppose I want to do develop a covariance-regularized regression
# method as in Section 3.2 of Witten and Tibshirani (2008). It will work
# like this:
# 1. Develop some positive definite estimate of Sigma
# 2. Find \beta by minimize \beta^T \Sigma \beta - 2 \beta^T X^T y +
# \lamda ||\beta||_1
# 3. Re-scale \beta.

# Step 1:
regcovx <- cov(x2)*(abs(cov(x2))>.005) + diag(ncol(x2))*.01

# Step 2:
betahat <-  crossProdLasso(regcovx, cov(x2,y), rho=.02)$beta
# Step 3:
betahat.sc <- betahat*lsfit(x2%*%betahat, y, intercept=FALSE)$coef
print(betahat.sc)

# Try a different value of rho:
betahat2 <- crossProdLasso(regcovx,cov(x2,y),rho=.04,beta.init=betahat)$beta
plot(betahat,betahat2, xlab="rho=.02",ylab="rho=.04")
#detach(diabetes)
```

`cv.scout` *Perform cross-validation for covariance-regularized regression, aka the Scout.*

Description

This function returns cross-validation error rates for a range of lambda1 and lambda2 values, and also makes beautiful CV plots if plot=TRUE.

Usage

```
cv.scout(x, y, K= 10,
  lam1s=seq(0.001,.2,len=10),lam2s=seq(0.001,.2,len=10),p1=2,p2=1,
  trace = TRUE, plot=TRUE,plotSE=FALSE,rescale=TRUE,...)
```

Arguments

`x`	A matrix of predictors, where the rows are the samples and the columns are the predictors
`y`	A matrix of observations, where length(y) should equal nrow(x)
`K`	Number of cross-validation folds to be performed; default is 10
`lam1s`	The (vector of) tuning parameters for regularization of the covariance matrix. Can be NULL if p1=NULL, since then no covariance regularization is taking place. If p1=1 and nrow(x)<ncol(x), then the no value in lam1s should be smaller than 1e-3, because this will cause graphical lasso to take too long. Also, if ncol(x)>500 then we really do not recommend using p1=1, as graphical lasso can be uncomfortably slow.
`lam2s`	The (vector of) tuning parameters for the L_1 regularization of the regression coefficients, using the regularized covariacne matrix. Can be NULL if p2=NULL. (If p2=NULL, then non-zero lam2s have no effect). A value of 0 will result in no regularization.
`p1`	The L_p penalty for the covariance regularization. Must be one of 1, 2, or NULL. NULL corresponds to no covariance regularization.
`p2`	The L_p penalty for the estimation of the regression coefficients based on the regularized covariance matrix. Must be one of 1 (for L_1 regularization) or NULL (for no regularization).
`trace`	Print out progress as we go? Default is TRUE.
`plot`	If TRUE (by default), makes beautiful CV plots.
`plotSE`	Should those beautiful CV plots also display std error bars for the CV? Default is FALSE
`rescale`	Scout rescales coefficients, by default, in order to avoid over-shrinkage
`...`	Additional parameters

Details

Pass in a data matrix x and a vector of outcomes y; it will perform (10-fold) cross-validation over a range of lambda1 and lambda2 values. By default, Scout(2,1) is performed.

Value

`folds`	The indices of the members of the K test sets are returned.
`cv`	A matrix of average cross-validation errors is returned.
`cv.error`	A matrix containing the standard errors of the elements in "cv", the matrix of average cross-validation errors.
`bestlam1`	Best value of lam1 found via cross-validation.
`bestlam2`	Best value fo lam2 found via cross-validation.
`lam1s`	Values of lam1 considered.
`lam2s`	Values of lam2 considered.

Author(s)

Daniela M. Witten and Robert Tibshirani

References

Witten, DM and Tibshirani, R (2008) Covariance-regularized regression and classification for high-dimensional problems. Journal of the Royal Statistical Society, Series B 71(3): 615-636. <http://www-stat.stanford.edu/~dwitten>

See Also

scout, predict.scoutobject

Examples

```
library(lars)
data(diabetes)
attach(diabetes)
par(mfrow=c(2,1))
par(mar=c(2,2,2,2))
## Not run: cv.sc <- cv.scout(x2,y,p1=2,p2=1)
## Not run: print(cv.sc)
## Not run: cv.la <- cv.lars(x2,y)
## Not run: print(c("Lars minimum CV is ", min(cv.la$cv)))
## Not run: print(c("Scout(2,1) minimum CV is ", min(cv.sc$cv)))
detach(diabetes)
```

`predict.scoutobject` *Prediction function for covariance-regularized regression, aka the Scout.*

Description

A function to perform prediction, using an x matrix and the output of the "scout" function.

Usage

```
## S3 method for class 'scoutobject'
predict(object, newx, ...)
```

Arguments

`object`	The results of a call to the "scout" function. The coefficients that are part of this object will be used for making predictions.
`newx`	The new x at which predictions should be made. Can be a vector of length ncol(x), where x is the data on which scout.obj was created, or a matrix with ncol(x) columns.
`...`	Additional arguments to predict

Value

`yhat`	If newx was a vector, then a matrix will be returned, with dimension length(lam1s)xlength(lam2s) (where lam1s and lam2s are attributes of scout.obj). The (i,j) element of this matrix will correspond to tuning parameter values (lam1s[i], lam2s[j]). If newx is a matrix, then an array of dimension nrow(newx)xlength(lam1s)xlength(lam2s) will be returned.

Author(s)

Daniela M. Witten and Robert Tibshirani

References

Witten, DM and Tibshirani, R (2008) Covariance-regularized regression and classification for high-dimensional problems. Journal of the Royal Statistical Society, Series B 71(3): 615-636. <http://www-stat.stanford.edu/~dwitten>

See Also

scout, cv.scout

Examples

```
library(lars)
data(diabetes)
attach(diabetes)
# Split data into training and test set
training <- sample(nrow(x2),floor(nrow(x2)/2))
xtrain <- x2[training,]
ytrain <- y[training]
xtest <- x2[-training,]
ytest <- y[-training]
# Done splitting data into training and test set
# Do cross-validation to determine best tuning parameter values for Scout(1,1)
## Not run: cv.out <- cv.scout(xtrain,ytrain,p1=1,p2=1, lam1s=seq(0.001,.15,len=5),K=4)
## Not run: print(cv.out)
# Done cross-validation
## Fit Model
#scout.object <- scout(xtrain,ytrain,p1=1,p2=1,lam1s=cv.out$bestlam1,lam2s=cv.out$bestlam2)
#print(scout.object)
## Done Fitting Model
## Predict on test data, and report MSE
#yhats <- predict(scout.object,xtest)
#print(mean((yhats-ytest)^2))
detach(diabetes)
```

---

print.cvobject *Print function for scout*

Description

A function to print CV output for scout

Usage

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

Arguments

x	The results of a call to the "cv.scout" function.
...	Additional arguments; ignored.

Author(s)

Daniela M. Witten and Robert Tibshirani

References

Witten, DM and Tibshirani, R (2008) Covariance-regularized regression and classification for high-dimensional problems. Journal of the Royal Statistical Society, Series B 71(3): 615-636.

See Also

scout, cv.scout

Examples

```
library(lars)
data(diabetes)
attach(diabetes)
# Split data into training and test set
training <- sample(nrow(x2),floor(nrow(x2)/2))
xtrain <- x2[training,]
ytrain <- y[training]
# Done splitting data into training and test set
# Do cross-validation to determine best tuning parameter values for Scout(1,1)
## Not run: cv.out <- cv.scout(xtrain,ytrain,p1=1,p2=1, lam1s=seq(0.001,0.1), K=4)
## Not run: print(cv.out)
# Done cross-validation
detach(diabetes)
```

---

print.scoutobject *Print function for scout*

Description

A function to print scout output

Usage

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

Arguments

x	The results of a call to the "scout" function.
...	additional arguments; these are ignored.

Author(s)

Daniela M. Witten and Robert Tibshirani

References

Witten, DM and Tibshirani, R (2008) Covariance-regularized regression and classification for high-dimensional problems. Journal of the Royal Statistical Society, Series B 71(3): 615-636. <http://www-stat.stanford.edu/~dwitten>

See Also

scout, cv.scout

Examples

```
library(lars)
data(diabetes)
attach(diabetes)
# Split data into training and test set
training <- sample(nrow(x2),floor(nrow(x2)/2))
xtrain <- x2[training,]
ytrain <- y[training]
xtest <- x2[-training,]
ytest <- y[-training]
# Done splitting data into training and test set
# Fit Model
scout.object <- scout(xtrain,ytrain,p1=1,p2=1,lam1s=c(0.001,0.1),lam2s=c(0.01,0.2))
print(scout.object)
# Done Fitting Model
detach(diabetes)
```

`scout` *Covariance-regularized regression, aka the Scout.*

Description

The main function of the "scout" package. Performs covariance-regularized regression. Required inputs are an x matrix of features (the columns are the features) and a y vector of observations. By default, Scout(2,1) is performed; however, p_1 and p_2 can be specified (in which case Scout(p_1, p_2) is performed). Also, by default Scout is performed over a grid of lambda1 and lambda2 values, but a different grid of values (or individual values, rather than an entire grid) can be specified.

Usage

```
scout(x,y,newx,p1=2,p2=1,lam1s=seq(.001,.2,len=10),lam2s=seq(.001,.2,len=10),
   rescale=TRUE, trace=TRUE,standardize=TRUE)
```

Arguments

x	A matrix of predictors, where the rows are the samples and the columns are the predictors
y	A matrix of observations, where length(y) should equal nrow(x)
newx	An *optional* argument, consisting of a matrix with ncol(x) columns, at which one wishes to make predictions for each (lam1,lam2) pair.
p1	The L_p penalty for the covariance regularization. Must be one of 1, 2, or NULL. NULL corresponds to no covariance regularization. WARNING: When p1=1, and ncol(x)>500, Scout can be SLOW. We recommend that for very large data sets, you use Scout with p1=2. Also, when ncol(x)>nrow(x) and p1=1, then very small values of lambda1 (lambda1 < 1e-4) will cause problems with graphical lasso, and so those values will be automatically increased to 1e-4.

`p2`	The L_p penalty for the estimation of the regression coefficients based on the regularized covariance matrix. Must be one of 1 (for L_1 regularization) or NULL (for no regularization).
`lam1s`	The (vector of) tuning parameters for regularization of the covariance matrix. Can be NULL if p1=NULL, since then no covariance regularization is taking place. If p1=1 and nrow(x)<ncol(x), then the no value in lam1s should be smaller than 1e-3, because this will cause graphical lasso to take too long. Also, if ncol(x)>500 then we really do not recommend using p1=1, as graphical lasso can be uncomfortably slow.
`lam2s`	The (vector of) tuning parameters for the L_1 regularization of the regression coefficients, using the regularized covariance matrix. Can be NULL if p2=NULL. (If p2=NULL, then non-zero lam2s have no effect). A value of 0 will result in no regularization.
`rescale`	Should coefficients beta obtained by covariance-regularized regression be rescaled by a constant, given by regressing y onto $x beta$? This is done in Witten and Tibshirani (2008) and is important for good performance. Default is TRUE.
`trace`	Print out progress? Prints out each time a lambda1 is completed. This is a good idea, especially when ncol(x) is large.
`standardize`	Should the columns of x be scaled to have standard deviation 1, and should y be scaled to have standard deviation 1, before covariance-regularized regression is performed? This affects the meaning of the penalties that are applied. In general, standardization should be performed. Default is TRUE.

Value

`intercepts`	Returns a matrix of intercepts, of dimension length(lam1s)xlength(lam2s)
`coefficients`	Returns an array of coefficients, of dimension length(lam1s)xlength(lam2s)xncol(x).
`p1`	p1 value used
`p2`	p2 value used
`lam1s`	lam1s used
`lam2s`	lam2s used

Note

When p1=1 and ncol(x)>500 or so, then Scout can be very slow!! Please use p1=2 when ncol(x) is large.

Author(s)

Daniela M. Witten and Robert Tibshirani

References

Witten, DM and Tibshirani, R (2008) Covariance-regularized regression and classification for high-dimensional problems. Journal of the Royal Statistical Society, Series B 71(3): 615-636. <http://www-stat.stanford.edu/~dwitten>

See Also

predict.scoutobject, cv.scout

Examples

```
library(lars)
data(diabetes)
attach(diabetes)
scout.out <- scout(x2,y,p1=2,p2=1)
print(scout.out)
detach(diabetes)
```

Index