Package 'LassoBacktracking'

April 4, 2017

Type Package

Title Modelling Interactions in High-Dimensional Data with
Backtracking

Version 0.1.2

Date 2017-04-03

Description Implementation of the algorithm introduced in Shah, R. D.
(2016) <http://www.jmlr.org/papers/volume17/13-515/13-515.pdf>.
Data with thousands of predictors can be handled. The algorithm
performs sequential Lasso fits on design matrices containing
increasing sets of candidate interactions. Previous fits are used to greatly
speed up subsequent fits so the algorithm is very efficient.

License GPL (>= 2)

Imports Matrix, parallel, Rcpp

LinkingTo Rcpp

URL www.jmlr.org/papers/volume17/13-515/13-515.pdf

RoxygenNote 5.0.1

NeedsCompilation yes

Author Rajen Shah [aut, cre]

Maintainer Rajen Shah <r.shah@statslab.cam.ac.uk>

Repository CRAN

Date/Publication 2017-04-04 08:48:43 UTC

R topics documented:

1

---

cvLassoBT   *Cross-validation for* LassoBT

Description

Perform k-fold cross-validation potentially multiple times on permuted version of the data.

Usage

```
cvLassoBT(x, y, lambda = NULL, nlambda = 100L,
  lambda.min.ratio = ifelse(nobs < nvars, 0.01, 1e-04), nfolds = 5L,
  nperms = 1L, mc.cores = 1L, ...)
```

Arguments

x	input matrix of dimension nobs by nvars; each row is an observation vector.
y	response variable; shoud be a numeric vector.
lambda	user supplied lambda sequence of decreasing penalty parameters. Typical usage is to allow the function to compute its own lambda sequence. Inappropraite sequences may cause convergence problems.
nlambda	the number of lambda values. Must be at least 3.
lambda.min.ratio	
	smallest value in lambda as a fraction of the largest value at which all main effects coefficients are 0.
nfolds	number of folds. Default is 5.
nperms	the number of permuted datasets to apply k-folds corss-validation to. Default is 1 so we carry out vanilla cross-validation.
mc.cores	the number of cores to use. Only applicable when not in Windows as it uses the **parallel** package to parallelise the computations.
...	other arguments that can be passed to LassoBT.

Value

A list with components as below.

lambda the sequence of lambda values used

cvm a matrix of error estimates (with squared error loss). The rows correspond to different lambda values whilst the columns correspond to different iterations

BT_fit a "BT" object from a fit to the full data.

cv_opt a two component vector giving the cross-validation optimal lambda index and iteration

cv_opt_err the minimal cross-validation error.

Examples

```
x <- matrix(rnorm(100*250), 100, 250)
y <- x[, 1] + x[, 2] - x[, 1]*x[, 2] + x[, 3] + rnorm(100)
out <- cvLassoBT(x, y, iter_max=10, nperms=2)
```

LassoBT	*Fit linear models with interactions using the Lasso.*

Description

Computes a number of Lasso solution paths with increasing numbers of interactions present in the design matrices corresponding to each path. Previous paths are used to speed up computation of subsequent paths so the process is very fast.

Usage

```
LassoBT(x, y, nlambda = 100L, iter_max = 1L,
  lambda.min.ratio = ifelse(nobs < nvars, 0.01, 1e-04), lambda = NULL,
  thresh = 1e-07, verbose = FALSE, inter_orig)
```

Arguments

x	input matrix of dimension nobs by nvars; each row is an observation vector.
y	response variable; shoud be a numeric vector.
nlambda	the number of lambda values. Must be at least 3.
iter_max	the number of iterations of the Backtracking algorithm to run. `iter_max=1` corresponds to a single lasso or elasticnet fit. Values greater than 1 will fit interactions.
lambda.min.ratio	
	smallest value in `lambda` as a fraction of the largest value at which all main effects coefficients are 0.
lambda	user supplied `lambda` sequence of decreasing penalty parameters. Typical usage is to allow the function to compute its own `lambda` sequence. Inappropraite sequences may cause convergence problems.
thresh	convergence threshold for coordinate descent. Each inner coordinate descent loop continues until either the maximum change in the objective after any coefficient update is less than `thresh` or `1E5` iterations have been performed.
verbose	if `TRUE` will print iteration numbers.
inter_orig	an optional 2-row matrix with each column giving interactions that are to be added to the design matrix before the algorithm begins.

Details

The Lasso optimisations are performed using coordinate descent similarly to the **glmnet** package. An intercept term is always included. Variables are centred and scaled to have equal empirical variance. Interactions are constructed from these centred and scaled variables, and the interactions themselves are also centred and scaled. Note the coefficients are returned on the original scale of the variables. Coefficients returned for interactions are for simple pointwise products of the original variables with no scaling.

Value

An object with S3 class "BT".

call the call that produced the object

a0 list of intercept vectors

beta list of matrices of coefficients stored in sparse column format (CsparseMatrix)

fitted list of fitted values

lambda the sequence of lambda values used

nobs the number of observations

nvars the number of variables

var_indices the indices of the non-constant columns of the design matrix

interactions a 2-row matrix with columns giving the interactions that were added to the design matrix

path_lookup a matrix with columns corresponding to iterations and rows to lambda values. Entry ij gives the component of the a0 and beta lists that gives the coefficients for the ith lambda value and jth iteration

l_start a vector with component entries giving the minimimum lambda index in the corresponding copmonents of beta and a0

References

Shah, R. D. (2016) *Modelling interactions in high-dimensional data with Backtracking. JMLR, to appear.* http://www.statslab.cam.ac.uk/~rds37/papers/shah16.pdf

See Also

predict.BT, coef.BT methods and the cvLassoBT function.

Examples

```
x <- matrix(rnorm(100*250), 100, 250)
y <- x[, 1] + x[, 2] - x[, 1]*x[, 2] + x[, 3] + rnorm(100)
out <- LassoBT(x, y, iter_max=10)
```

| predict.BT | *Make predictions from a "BT" object.* |

Description

Similar to other predict methods, this function predicts fitted values and computes coefficients from a fitted "BT" object.

Usage

```
## S3 method for class 'BT'
predict(object, newx, s = NULL, iter = NULL,
  type = c("response", "coefficients"), ...)

## S3 method for class 'BT'
coef(object, s = NULL, iter = NULL, ...)
```

Arguments

object	fitted "BT" object.
newx	matrix of new values of design matrix at which predictions are to be made. Ignored when `type=="coefficients"`.
s	value of the penalty parameter at which predictions are required. If the value is not one of the `lambda` values present in `object` the output will be etermined by linear interpolation. Default is the entire sequence of `lambda` values present in `object`.
iter	iteration at which predictions are required. Default is the entire sequence of iterations in `object`.
type	of prediction required. Type `"response"` gives estimates of the response whilst type `"coefficients"` gives coefficient estimates.
...	not used. Other arguments to `predict`.

Value

Either a vector of predictions or, if either `s` or `iter` are `NULL`, a three-dimensional array with last two dimensions indexing different `lambda` values and iterations.

Examples

```
x <- matrix(rnorm(100*250), 100, 250)
y <- x[, 1] + x[, 2] - x[, 1]*x[, 2] + x[, 3] + rnorm(100)
out <- LassoBT(x, y, iter_max=10)
predict(out, newx=x[1:2, ])
```

Index