Package ‘MetaLasso’

March 22, 2018

Type Package
Title Integrative Generalized Linear Model for Group/Variable Selections over Multiple Studies
Version 0.1.0
Depends glmnet
Author Quefeng Li
Maintainer Quefeng Li <quefeng@email.unc.edu>

Description
A flexible variable selection tool that selects variables and groups of variables from multiple studies. It was built for a high-dimensional generalized linear model integrating data from multiple studies. An application of this tool is to select genes and pathways from multiple genomic data with various response types. For more details, see the reference below.

License GPL-2

Reference

RoxygenNote 6.0.1

R topics documented:

grpmetlasso ................................................................. 1
metlasso ................................................................. 3

Index 6

---

grpmetlasso

Solve the group MetaLasso problem with a single tuning parameter

Description
Jointly fit a generalized linear model with a group penalty over multiple datasets. It enables both group selections and within-group variable selections over multiple datasets. Fits linear, logistic and multinomial, poisson, and Cox regression models.
Usage

```r
grpmetalasso(X.all, Y.all, obs, groups, lambda, family = c("gaussian", "binomial", "poisson", "multinomial", "cox", "mgaussian"), maxit = 100, tol = 0.001)
```

Arguments

- `X.all`: a concatenated design matrix, of dimension `nobs*nvars`, where `nobs` is the total sample size over multiple datasets and `nvars` is the total number of variables.
- `Y.all`: a concatenated response vector from all datasets
- `obs`: a vector of sample sizes of multiple datasets
- `groups`: a matrix, of dimension `ngrps*nvars`, indexing the group membership of variables. The `(i, j)`-th element of `groups = 1`, if the `j`-th variable belongs to the `i`-th group; = 0, otherwise. A variable is allowed to belong to multiple groups.
- `lambda`: a tuning parameter of penalty
- `family`: response type (see above)
- `maxit`: maximal number of iterations allowed
- `tol`: tolerance level of convergence

Details

The function minimizes $-\logLik + \lambda * p(\beta)$, where $-\logLik$ is the negative of the total log-Likelihood from all datasets, $\lambda$ is a single tuning parameter and $p(\beta)$ is a specific group penalty function enabling both group selections and within-group variable selections over multiple datasets. For more details of the penalty function, see the reference below.

Value

A list of following components

- `coe`: estimated coefficients in each dataset
- `grp.coef`: estimated group effects. For more details, see the reference below.
- `iteration`: number of iterations
- `converge`: TRUE if convergence is achieved
- `diff`: last step difference

References


Examples

```r
dign <- function(m1, m2){
  rbind(cbind(m1, matrix(rep(0, nrow(m1)*ncol(m2))), nrow = nrow(m1))),
    cbind(matrix(rep(0, nrow(m2)*ncol(m1))), nrow = nrow(m2), m2))
}

M <- 10  # number of datasets
n.m <- rep(50, M)  # number of n.m in each dataset
p <- 100  # number of covariates
```
metalasso <- p/5  # number of pathways
nonzero <- 25  # number of nonzero coefficients
means <- c(rep(8, 5), rep(8, 5),
            rep(-4, 5), rep(-4, 5), rep(-8, 5),
            rep(0, p - nonzero))  # means of nonzero beta's
sig <- c(rep(0.5, nonzero), rep(0, p - nonzero))  # sds of nonzero beta's

groups <- matrix(rep(1, 5), nrow = 1)  # group structure
for (i in 1:(K-1)) {
  groups <- diag(groups, matrix(rep(1, 5), nrow = 1))
}

## generate beta
beta <- NULL
for (i in 1:p){
  beta <- cbind(beta, rnorm(M, means[i], sig[i]))
}

## generate X.all and Y.all
X.all <- NULL
Y.all <- NULL
for (m in 1:M){
  X.tmp <- matrix(scale(matrix(rnorm(n.m[m] * p), n.m[m], p)), n.m[m], p)
  X.all <- rbind(X.all, X.tmp)
  pb <- X.tmp %*% beta[m, ]
  pb <- exp(pb) / (1 + exp(pb))
  Y.tmp <- matrix(rbinom(n.m[m], 1, pb), ncol = 1)
  Y.all <- rbind(Y.all, Y.tmp)
}
Y.all <- as.vector(Y.all)

## range of tuning parameters
lams <- 2^seq(-3, -1, len = 10)

BIC <- NULL
for (i in 1:length(lams)){
  fit <- grpmetalasso(X.all, Y.all, obs = n.m, groups = groups, family = 'binomial', lambda = lams[i])
  BIC[i] <- bic(X.all, Y.all, n.m, fit$coe)
}

best.fit <- grpmetalasso(X.all, Y.all, obs = n.m, groups = groups, family = 'binomial',
                         lambda = which.min(BIC))

metalasso  # Solve the MetaLasso problem with a single tuning parameter

Description

Jointly fit a generalized linear model with a penalty over multiple datasets. It enables heterogeneous variable selections in different datasets. Fits linear, logistic and multinomial, poisson, and Cox regression models.

Usage

metalasso(X.all, Y.all, obs, lambda, family = c("gaussian", "binomial",
"poisson", "multinomial", "cox", "mgaussian"), maxit = 100, tol = 0.001)
Arguments

X.all a concatenated design matrix, of dimension nobs*nvars, where nobs is the total sample size over multiple datasets and nvars is the total number of variables.

Y.all a concatenated response vector from all datasets

obs a vector of sample sizes of multiple datasets

lambda a tuning parameter of penalty

family response type (see above)

maxit maximal number of iterations allowed

tol tolerance level of convergence

Details

The function minimizes \(-logLik + lambda * p(beta)\), where \(-logLik\) is the negative of the total log-Likelihood from all datasets, \(lambda\) is a single tuning parameter and \(p(beta)\) is a specific penalty function enabling heterogeneous selections of variables in different datasets. For more details of the penalty function, see the reference below.

Value

a list of the following components

coe estimated coefficients in each dataset

iteration number of iterations

converge TRUE if convergence is achieved

diff last step difference

References


Examples

n <- 50
p <- 100
M <- 5
obs <- rep(n, M)

X.all <- NULL
Y.all <- NULL

for (m in 1:M) {
  X.tmp <- matrix(scale(matrix(rnorm(obs[m] * p), obs[m], p)), obs[m], p)
  X.all <- rbind(X.all, X.tmp)
  beta <- c(1, -1, 2, -1, rep(0, p - 4))
  pb <- X.tmp %*% beta
  pb <- exp(pb) / (1 + exp(pb))
  Y.tmp <- matrix(rbinom(obs[m], 1, pb), ncol = 1)
  Y.all <- c(Y.all, Y.tmp)
}

lams <- seq(0.01, 0.08, len = 10)
BIC <- NULL
for (j in 1:length(lams)) {
  fit <- metalasso(X.all, Y.all, obs, family = 'binomial', lambda = lams[j])
  BIC[j] <- bic(X.all, Y.all, obs, fit$coe)
}

best.fit <- metalasso(X.all, Y.all, obs, lambda = lams[which(BIC == min(BIC))], family = 'binomial')
Index

grpmetalasso, 1
metalasso, 3