Package 'glmaag'

May 10, 2019

Title Adaptive LASSO and Network Regularized Generalized Linear Models

Version 0.0.6

Date 2019-05-09

Author Kaiqiao Li [aut, cre],
Pei Fen Kuan [aut],
Xuefeng Wang [aut]

Maintainer Kaiqiao Li <kaiqiao.li@stonybrook.edu>

Description Efficient procedures for adaptive LASSO and network regularized for Gaussian, logistic, and Cox model. Provides network estimation procedure (combination of methods proposed by Ucar, et. al (2007) <doi:10.1093/bioinformatics/btm423> and Meinshausen and Buhlmann (2006) <doi:10.1214/009053606000000281>), cross validation and stability selection proposed by Meinshausen and Buhlmann (2010) <doi:10.1111/j.1467-9868.2010.00740.x> and Liu, Roeder and Wasserman (2010) <arXiv:1006.3316> methods. Interactive R app is available.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.6.0), survival, data.table

Imports Rcpp (>= 1.0.0), methods, stats, Matrix, ggplot2, gridExtra, maxstat, survminer, plotROC, shiny, foreach, pROC, huge, OptimalCutpoints

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-05-10 07:50:16 UTC

R topics documented:

`coef.cv_glmaag` *Coefficients*

Description

Get the coefficients estimated by the cv_glmaag model

Usage

```
## S3 method for class 'cv_glmaag'
coef(object, type1se = T, ...)
```

Arguments

`object`	the estimated cv_glmaag model
`type1se`	whether or not used 1 SE error (default to be TRUE)
`...`	...

Value

estimated coefficient included intercept (Cox model does not return intercept)

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
cvwhich <- sample(rep(0:4, length.out = length(y)))
mod <- cv_glmaag(y, x, L0, cvwhich = cvwhich)
cc <- coef(mod)
```

`coef.glmaag` *Coefficients for glmaag*

Description

Get coefficients for glmaag objects

Usage

```
## S3 method for class 'glmaag'
coef(object, lam1, lam2, ...)
```

Arguments

`object`	fitted glmaag object
`lam1`	lambda1 sequence need coefficients, must be within the fitted model
`lam2`	lambda2 sequence need coefficients, must be within the fitted model
`...`	...

Value

coefficients

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
mod <- glmaag(y, x, L0)
cc <- coef(mod)
```

coef.ss_glmaag *Coefficients for ss_glmaag*

Description

Get the coefficients tuned by stability selection

Usage

```
## S3 method for class 'ss_glmaag'
coef(object, ...)
```

Arguments

object	the model estimated via stability selection
...	...

Value

the optimal coefficients get from stability selection including intercept (except for Cox)

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, 4:6]
mod <- ss_glmaag(y, x, L0[seq_len(3), seq_len(3)], nsam = 3)
cc <- coef(mod)
```

cv_glmaag *Cross validation for glmaag*

Description

Do k-fold cross-validation for glmaag

Usage

```
cv_glmaag(y, x, L, nfolds = 5, cvwhich, foldseed, stratify = T,
  gam = 1, tune = F, est = T, lam1, lam2, dfmax, w0, adaptl1 = T,
  adaptl2 = T, pind, intercept = T, standardize = T,
  maxiter = 10000, cri = 0.001, fam = "Gaussian", measdev = T,
  type1se = T, parallel = F)
```

Arguments

y	outcome
x	predictors matrix
L	Laplacian matrix for the first network
nfolds	number of folds used in cross validation, default to be five
cvwhich	fold assignment, start from zero, if missing do random cross validation
foldseed	the random seed for cross validation design
stratify	whether to do stratified cross validation for Logistic or Cox model, default to be TRUE
gam	The power of weights of L1 penalty, default to be ones
tune	whether to tune the input network with estimated network or identity matrix, ignored if no input network
est	when there is no input network whether to use estimated network or identiy matrix (elastic net) or mixed the network with estimated network or identity matrix, default to be estimated network
lam1	The tuning parameters for L1 penalty. If not defined, searched by default
lam2	The tuning parameters for quadratic penalty. If not defined, searched by default
dfmax	maximum number of parameters allowed in the model, default to be p/2
w0	Weights for L1 penalty. If not defined, estimated via quadratic penalyzed regression
adaptl1	whether to adapt the L1 penalty, default to be TRUE
adaptl2	whether to adapt the sign for quadratic penalty, default to be TRUE
pind	indicator vector whether to put L1 penalty on the feature, 1 means penalyzed while 0 means not penalyzed, default to be all ones (all penalyzed)
intercept	whether to include intercept. Ignore for Cox regression
standardize	whether to standardize predictors
maxiter	maximum number of iterations, default to be 500
cri	stoppint criterion, default to be 0.001
fam	family for the outcome, can be "Gaussian", "Logistic", and "Cox"
measdev	Whether to use deviance to tune, default to be deviance. If not, use mean absolue error, area under ROC curve, or concordance index for Gaussian, Logistic, and Cox
type1se	whether to use one standard error or maximum rule, default to be one standard error rule
parallel	whether to do parallel computing at each fold, need to set up parallel first, default to be FALSE

Value

input	input predictor matrix
inputweight	estimated weights if mixing network
lambda1	lambda1 path that has been searched
lambda1	lambda1 path that has been searched
lambda1_max	selected lambda1 based on maximum rule
lambda2_max	selected lambda2 based on maximum rule
lambda1_1se	selected lambda1 based on one standard error rule
lambda2_1se	selected lambda2 based on one standard error rule
cvm	the mean cross validation accuracy
cv1se	the standard error of cross validation accuracy
cvn	the mean number of parameter estimated among folds
n_max	number of selected features based on maximum rule
n_1se	number of selected features based on one standard error rule
intercept_max	estimated intercept based on maximum rule
intercept_1se	estimated intercept based on one standard error rule
coef_max	estimated coefficients based on maximum rule
coef_1se	estimated coefficients based on one standard error rule
fam	family of outcome
measure	measure in cross validation

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
cvwhich <- sample(rep(0:4, length.out = length(y)))
mod <- cv_glmaag(y, x, L0, cvwhich = cvwhich)
```

---

evaluate *Evaluate prediction*

Description

Evaluate goodness of prediction.

Usage

```
evaluate(y_pre, y, cutpoint = 0.5, fam = "Gaussian")
```

Arguments

y_pre	predicted value
y	actual values (class for binary phenotype and Surv object for right censored phenotype)
cutpoint	cutpoints for binary phenotype, default to be 0.5
fam	family of the phenotype, can be "continous", "binary", or "Cox"

Value

goodness of prediction

Examples

```
x <- rnorm(100)
y <- rnorm(100)
evaluate(x, y)
```

---

evaluate_plot *Prediction visualization*

Description

Sample plots for prediction evaluation (scatter plot for Gaussian, ROC curve for logistic, and Kaplan Meier curve for Cox)

Usage

```
evaluate_plot(y_pre, y_test, fam = "Gaussian", mod, y_train, cutp)
```

Arguments

y_pre	predicted value
y_test	actual value
fam	type of predicted outcome, can be "Gaussian" (default), "Logisitc", and "Cox"
mod	fitted glmagarph model, must be available for Cox if cutpoint not provided
y_train	the training outcome to obtain
cutp	cutpoint for Cox model

Value

plots

Examples

```
x <- rnorm(100)
y <- x + rnorm(100)
evaluate_plot(x, y)
```

getcut *Get optimal cut points for binary or right censored phenotype*

Description

Obtain optimal cut point based on Youden index for binary phenotype and log rank test for right censored phenotype.

Usage

```
getcut(pre, act, fam = "Logistic")
```

Arguments

pre	predicted value
act	actual values (class for binary phenotype and Surv object for right censored phenotype)
fam	the family of the outcome, can be "Gaussian", "Logistic" or "Cox"

Value

optimal cut point

Examples

```
x <- rnorm(100)
y <- as.numeric(x + rlogis(100) > 0)
getcut(x, y)
```

getS *Estimate standardized Laplacian matrix*

Description

Estimate standardized Laplacian matrix given data using gene co-expression network method

Usage

```
getS(x, sparse = T)
```

Arguments

x	data
sparse	estimate a sparse network or not, default to be T, but may be slow

Value

standardized laplacian matrix

References

Ucar D, Neuhaus I, Ross-MacDonald P, Tilford C, Parthasarathy S, et al. (2007) Construction of a reference gene association network from multiple profiling data: application to data analysis. Bioinformatics 23: 2716-2724.

Meinshausen, N., & B"uhlmann, P. (2006). High-dimensional graphs and variable selection with the lasso. The annals of statistics, 1436-1462.

Examples

```
xx <- matrix(rnorm(12), 3, 4)
ss <- getS(xx, FALSE)
```

`glmaag` *Fit glmaag model*

Description

Fit the glmaag model with given tuning parameters without cross validation or stability selection

Usage

```
glmaag(y, x, L, tune = F, est = T, gam = 1, lam1, lam2, nfolds = 5,
  dfmax, w0, adaptl1 = T, adaptl2 = T, pind, intercept = T,
  standardize = T, maxiter = 10000, cri = 0.001, fam = "Gaussian",
  measdev = T, type1se = T, parallel = F)
```

Arguments

`y`	outcome
`x`	predictors matrix
`L`	Laplacian matrix for the network
`tune`	whether to tune with an estimated network, default to be FALSE
`est`	whether to estimate a network from the data
`gam`	the parameter for l1 adaptive weight, default to be ones
`lam1`	The tuning parameters for L1 penalty. If not defined, searched by default
`lam2`	The tuning parameters for quadratic penalty. If not defined, searched by default
`nfolds`	number of folds used in cross validation to obatin network sign estimate and l1 weight estimate, default to be five
`dfmax`	maximum number of parameters allowed in the model, default to be p/2

w0	Weights for l1 penalty. If not defined, estimated via quadratic penalyzed regression
adaptl1	whether to adapt the l1 penalty, default to be TRUE
adaptl2	whether to adapt the sign for quadratic penalty, default to be TRUE
pind	indicator vector whether to put L1 penalty on the feature, 1 means penalyzed while 0 means not penalyzed, default to be all ones (all penalyzed)
intercept	whether to include intercept. Ignore for Cox regression
standardize	whether to standardize predictors
maxiter	maximum number of iterations, default to be 500
cri	stoppint criterion, default to be 0.001
fam	family for the outcome, can be "Gaussian", "Logistic", and "Cox"
measdev	Whether to use deviance to tune when estimate l1 weight and network sign, default to be deviance. If not, use mean absolue error, area under ROC curve, or concordance index for Gaussian, Logistic, and Cox
type1se	whether to use one standard error or maximum rule when estimate network sign and l1 weight, default to be one standard error rule
parallel	whether to do parallel computing at each lambda2, need to set up parallel first, default to be FALSE

Value

input	input predictors
lambda1	l1 penalty parameter search sequence
lambda2	quadratic penalty parameter search sequence
ns	number of parameters selected given provided tuning parameter
coefs	coefficients estimated
intercept	intercepts estimated
loglik	log likelihood estimated
fam	family of the outcome

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
mod <- glmaag(y, x, L0)
```

L0 *sample network 0*

Description

A Laplacian matrix for the predictors

Usage

```
L0
```

Format

a matrix with 20 rows and 20 columns

L1 *sample network 1*

Description

An alternative Laplacian matrix for the predictors

Usage

```
L1
```

Format

a matrix with 20 rows and 20 columns

laps *Standardized Laplacian matrix*

Description

Obtain standardized Laplacian matrix given adjacency matrix

Usage

```
laps(A)
```

Arguments

A	adjacency matrix

Value

Laplacian matrix

Examples

```
a <- matrix(0, 2, 2)
la <- laps(a)
```

---

plot.cv_glmaag *Cross validation plot*

Description

plot cross validation performance paths

Usage

```
## S3 method for class 'cv_glmaag'
plot(x, col_count = 3, SE = T, ...)
```

Arguments

x	the cv_glmaag object
col_count	number of columns in the plots
SE	whether or not plot the standared error curves (when SE = TRUE)
...	...

Value

plot generated by the model

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
cvwhich <- sample(rep(0:4, length.out = length(y)))
mod <- cv_glmaag(y, x, L0, cvwhich = cvwhich)
gg <- plot(mod, SE = FALSE)
```

plot.glmaag *Paths for glmaag object*

Description

Generates coefficients, lok likelihood, or number of parameters paths for glmaag models

Usage

```
## S3 method for class 'glmaag'
plot(x, col_count = 3, type = "coef", ...)
```

Arguments

x	glmaag object
col_count	number of columns shown in the plot (when type = 'coef)
type	can be "coef" (coefficients paths), "loglik" (log likelihood paths), or "n" (number of parameters paths)
...	...

Value

plots

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
mod <- glmaag(y, x, L0)
gg <- plot(mod, type = 'loglik')
```

plot.ss_glmaag *Instability plot*

Description

Instability path plot

Usage

```
## S3 method for class 'ss_glmaag'
plot(x, ...)
```

Arguments

x	the input ss_glmagrph object
...	...

Value

the instability path

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, 4:6]
mod <- ss_glmaag(y, x, L0[seq_len(3), seq_len(3)], nsam = 3)
gg <- plot(mod)
```

---

predict.cv_glmaag *Predict*

Description

Prediction for cv_glmaag model

Usage

```
## S3 method for class 'cv_glmaag'
predict(object, x, type1se = T, type = "link", ...)
```

Arguments

object	the estimated cv_glmaag model
x	the new dataset for prediction, if omitted returns the training prediction
type1se	whether or not using the coefficients by one standard error ruld, default to be TRUE
type	can be either "link", or "response", link returns linear predicted score, For Gaussian model this option can be ingnored, for logistic model "response" returns predicted probability, for Cox model "reponse" returns relative risk
...	...

Value

the predicted value

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
cvwhich <- sample(rep(0:4, length.out = length(y)))
mod <- cv_glmaag(y, x, L0, cvwhich = cvwhich)
pp <- predict(mod)
```

`predict.glmaag` *Prediction for glmaag*

Description

Prediction using glmaag model

Usage

```
## S3 method for class 'glmaag'
predict(object, x, lam1, lam2, type = "link",
  cutp = 0.5, ...)
```

Arguments

`object`	fitted glmaag object
`x`	The new dataset to be predicted, do training prediction if x is missing
`lam1`	lambda1 sequence for prediction, must be within the fitted model
`lam2`	lambda2 sequence for prediction, must be within the fitted model
`type`	type of prediction (can be "link", "reponse"), ignored for Gaussian model. "link" is the linear predicted score, "response" is the predicted probability for logistic model and relative risk for Cox model
`cutp`	the cut off value for binary outcome, default to be 0.5
`...`	...

Value

predicted values

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
mod <- glmaag(y, x, L0)
pp <- predict(mod)
```

predict.ss_glmaag *Prediction via stability selection*

Description

Predict using the model tuned by stability selection

Usage

```
## S3 method for class 'ss_glmaag'
predict(object, x, type = "link", ...)
```

Arguments

object	the ss_glmaag object
x	the new dataset to be predicted, do training prediction if x is missing
type	type of prediction (can be "link", or "reponse"), ignored for Gaussian model. "link" is the linear predicted score, "response" is the predicted probability for logistic model and relative risk for Cox model
...	...

Value

the predicted values

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, 4:6]
mod <- ss_glmaag(y, x, L0[seq_len(3), seq_len(3)], nsam = 3)
pp <- predict(mod)
```

print.cv_glmaag *the results of the cross validation model*

Description

print fitted information

Usage

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

Arguments

x	the fitted cv_glmaag object
...	...

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
cvwhich <- sample(rep(0:4, length.out = length(y)))
mod <- cv_glmaag(y, x, L0, cvwhich = cvwhich)
print(mod)
```

---

print.ss_glmaag *the results of the stability selection model*

Description

print fitted information

Usage

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

Arguments

x	the fitted ss_glmaag object
...	...

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, 4:6]
mod <- ss_glmaag(y, x, L0[seq_len(3), seq_len(3)], nsam = 3)
print(mod)
```

runtheExample *Shiny app*

Description

Run the shiny app for web interative using. You need to load data.table, dplyr, ggplot2, plotROC, and survminer beforehand. If you want to do parallel computing, you also need to register cores.

Usage

```
runtheExample(whetherrun)
```

Arguments

whetherrun	whether to run shiny app, default to be TRUE

Examples

```
runtheExample(FALSE)
```

sampledata *Simulated data*

Description

A data set containing outcome for Gaussian, logistic, and Cox variables and 20 predictors

Usage

```
sampledata
```

Format

a data frame with 100 rows and 23 variables

ss_glmaag *Stability selection for glmaag*

Description

Do stability selection for glmaag

Usage

```
ss_glmaag(y, x, L, nfolds = 5, subn, nsam = 100, beta = 0.15,
  gam = 1, tune = F, est = T, lam1, lam2, w0, adaptl1 = T,
  adaptl2 = T, pind, intercept = T, standardize = T,
  maxiter = 10000, cri = 0.001, fam = "Gaussian", measdev = T,
  type1se = T, parallel = F)
```

Arguments

y	outcome
x	predictors matrix
L	Laplacian matrix for the first network
nfolds	number of folds used in cross validation to estimate the l1 weights or network tuning, default to be five
subn	number of samples in each subset, default to be n/2 if n<400 and 10sqrt(10) if n>400
nsam	number of subsets, default to be 100
beta	the cut off for instability score
gam	The power of weights of l1 penalty, default to be ones
tune	whether to tune the input network with estimated network or identity matrix, ignored if no input network
est	when there is no input network whether to use estimated network or identiy matrix (elastic net) or mixed the network with estimated network or identity matrix, default to be estimated network
lam1	The tuning parameters for l1 penalty. If not defined, searched by default
lam2	The tuning parameters for quadratic penalty. If not defined, searched by default
w0	Weights for l1 penalty. If not defined, estimated via quadratic penalyzed regression
adaptl1	whether to adapt the l1 penalty, default to be TRUE
adaptl2	whether to adapt the sign for quadratic penalty, default to be TRUE
pind	indicator vector whether to put l1 penalty on the feature, 1 means penalyzed while 0 means not penalyzed, default to be all ones (all penalyzed)
intercept	whether to include intercept. Ignore for Cox regression
standardize	whether to standardize predictors

maxiter	maximum number of iterations, default to be 500
cri	stoppint criterion, default to be 0.001
fam	family for the outcome, can be "Gaussian", "Logistic", and "Cox"
measdev	Whether to use deviance to tune when estimate l1 weight and network sign, default to be deviance. If not, use mean absolue error, area under ROC curve, or concordance index for Gaussian, Logistic, and Cox
type1se	whether to use one standard error or maximum rule for l1 weight estimation and network sign, default to be one standard error rule
parallel	whether to do parallel computing at each subset, need to set up parallel first, default to be FALSE

Value

input	input matrix for predictors
lambda1	searching sequence for l1 penalty parameters
lamdba2	searching sequence for quadratic penalty parameters
lambda1_ss	optimal l1 parameter
lambda2_ss	optimal quadratic parameter
n_ss	number of parameters obtained by the optimal model
ssm	instability score paths
ssf	selection probability paths
intercept_ss	intercept estimated by the optimal model
coef_ss	coefficients estimated by the optimal model
fam	the family of the outcome

References

Meinshausen, N., & B"uhlmann, P. (2010). Stability selection. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 72(4), 417-473.

Liu, H., Roeder, K., & Wasserman, L. (2010). Stability approach to regularization selection (stars) for high dimensional graphical models. In Advances in neural information processing systems (pp. 1432-1440).

Examples

```
data(sampledata)
data(L0)
y <- sampledata$Y_Gau
x <- sampledata[, 4:6]
mod <- ss_glmaag(y, x, L0[seq_len(3), seq_len(3)], nsam = 3)
```

tune_network *tune two network*

Description

Tune two network for better prediction.

Usage

```
tune_network(y, x, L1, L2, adaptl2 = T, nfolds = 5, cvwhich, foldseed,
  stratify = T, lam0, bets, intercept = T, standardize = T,
  fam = "Gaussian", type1se = T, measdev = T, maxiter = 10000,
  cri = 0.001, parallel = F)
```

Arguments

y	outcome
x	predictors matrix
L1	Laplacian matrix for the first network
L2	Laplacian matrix for the second network
adaptl2	whether to adapt the sign for quadratic penalty, default to be TRUE
nfolds	number of folds used in cross validation, default to be five
cvwhich	fold assignment, start from zero, if missing do random cross validation
foldseed	the random seed for cross validation design
stratify	whether to do stratified cross validation for Logistic or Cox model, default to be TRUE
lam0	The tuning parameters for quadratic penalty. If not defined, tuned by default
bets	The candidate weight for the first network, must be between 0 and 1, default to be 0, 0.1,..., 1
intercept	whether to include intercept. Ignore for Cox regression
standardize	whether to standardize predictors
fam	family for the outcome, can be "Gaussian", "Logistic", and "Cox"
type1se	whether to use one standard error or maximum rule, default to be one standard error rule
measdev	Whether to use deviance to tune, default to be deviance. If not, use mean absolue error, area under ROC curve, or concordance index for Gaussian, Logistic, and Cox
maxiter	maximum number of iterations, default to be 500
cri	stoppint criterion, default to be 0.001
parallel	whether to do parallel computing at each fold

Value

est	estimated mixed Laplacian matrix
weight	weights for the two Laplacian matrix

Examples

```
data(sampledata)
data(L0)
data(L1)
y <- sampledata$Y_Gau
x <- sampledata[, -(1:3)]
Ltune <- tune_network(y, x, L0, L1, adaptl2 = FALSE)
weight <- Ltune@weight
Lest <- Ltune@est
```

Index