

Package ‘gren’

October 13, 2022

Type Package

Title Adaptive Group-Regularized Logistic Elastic Net Regression

Version 0.0.1

Date 2018-07-19

Author Magnus M. Münch

Maintainer Magnus M. Münch <m.munch@vumc.nl>

Description Allows the user to incorporate multiple sources of co-data (e.g., previously obtained p-values, published gene lists, and annotation) in the estimation of a logistic regression model to enhance predictive performance and feature selection, as described in Münch, Peeters, van der Vaart, and van de Wiel (2018) <[arXiv:1805.00389](#)>.

License GPL (>= 2)

Imports Rcpp (>= 0.12.14), glmnet, Iso, pROC

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

VignetteBuilder knitr

URL <https://github.com/magnusmunch/gren/>

Suggests knitr, rmarkdown

Encoding UTF-8

RoxygenNote 6.0.1

LazyData true

Repository CRAN

Date/Publication 2018-07-30 13:10:03 UTC

R topics documented:

gren-package	2
cv.gren	3
dataCervical	5
dataColon	6

denet	7
gren	8
predict.gren	11

Index	13
--------------	-----------

gren-package	<i>Adaptive group-regularized logistic elastic net regression</i>
--------------	---

Description

The package allows the user to incorporate multiple sources of co-data (e.g., previously obtained p-values, published gene lists, and annotation) in the estimation of a logistic regression model to enhance predictive performance.

Details

The main function of the package is [gren](#), which estimates a group-regularized elastic net regression model. The following functions are convenience functions:

[cv.gren](#) estimates performance measures by efficient cross-validation.

[coef.gren](#) S3 method to retrieve model parameters from a [gren](#) fit.

[predict.gren](#) S3 method to get predictions for new data from a [gren](#) fit.

[denet](#) density function of the elastic net prior distribution.

[renet](#) generate samples from the elastic net prior distribution.

Author(s)

Magnus M. Münch Maintainer: Magnus M. Münch <m.munch@vumc.nl>

References

Münch, M.M., Peeters, C.F.W., van der Vaart, A.W., and van de Wiel, M.A. (2018). Adaptive group-regularized logistic elastic net regression. arXiv:1805.00389v1 [stat.ME].

See Also

[cv.gren](#)

Examples

```
## Create data
p <- 1000
n <- 100
set.seed(2018)
x <- matrix(rnorm(n*p), ncol=p, nrow=n)
beta <- c(rnorm(p/2, 0, 0.1), rnorm(p/2, 0, 1))
```

```

m <- rep(1, n)
y <- rbinom(n, m, as.numeric(1/(1 + exp(-x %*% as.matrix(beta)))))
partitions <- list(groups=rep(c(1, 2), each=p/2))

## estimate model
fit.gren <- gren(x, y, m, partitions=partitions)

```

cv.gren	<i>Performance cross-validation of group-regularized logistic elastic net regression</i>
---------	--

Description

Function that cross-validations for performance estimation of gren models.

Usage

```

cv.gren(x, y, m=rep(1, nrow(x)), unpenalized=NULL, partitions=NULL, alpha=0.5,
lambda=NULL, intercept=TRUE, monotone=NULL, psel=TRUE, compare=TRUE,
posterior=FALSE, nfolds=nrow(x), foldid=NULL, trace=TRUE,
control=list(epsilon=0.001, maxit=500, maxit.opt=1000, maxit.vb=100),
keep.pred=TRUE, fix.lambda=FALSE, nfolds.out=nrow(x), foldid.out=NULL,
type.measure=c("auc", "deviance", "class.error"))

```

Arguments

x	See gren .
y	See gren .
m	See gren .
unpenalized	See gren .
partitions	See gren .
alpha	See gren .
lambda	See gren .
intercept	See gren .
monotone	See gren .
psel	See gren .
compare	See gren .
posterior	See gren .
nfolds	See gren .
foldid	See gren .
trace	if TRUE, progress of the cross-validation is printed.
control	See gren .

keep.pred	logical. If TRUE the cross-validated predictions are saved.
fix.lambda	logical. If TRUE lambda is cross-validated only once and used in every fold.
nfolds.out	numeric that gives the number of folds to use in the performance cross-validation. Default is nrow(x).
foldid.out	optional numeric vector of length nrow(x) with the performance cross-validation fold assignments of the observations.
type.measure	numeric that gives the performance measures to calculate. Currently possible are AUC, deviance, and misclassification error.

Details

cv.gren is a convenience function that gives cross-validated predictions. Performance measures are optionally calculated with these predictions. cv.gren is more efficient than simply looping over the folds, since it uses the final estimates of the previous fold as starting values for the next fold. This substantially reduces computation time.

Value

Function returns a list of length two with the following components:

groupreg	list with the cross-validated predictions of length nrow(x) and possibly performance metrics of the group-regularized model.
regular	list with the cross-validated predictions of length nrow(x) and possibly performance metrics of the regular model.

Author(s)

Magnus M. Münch <m.munch@vumc.nl>

References

Münch, M.M., Peeters, C.F.W., van der Vaart, A.W., and van de Wiel, M.A. (2018). Adaptive group-regularized logistic elastic net regression. arXiv:1805.00389v1 [stat.ME].

See Also

[gren](#), [predict.gren](#), [coef.gren](#)

Examples

```
## Create data
p <- 1000
n <- 100
set.seed(2018)
x <- matrix(rnorm(n*p), ncol=p, nrow=n)
beta <- c(rnorm(p/2, 0, 0.1), rnorm(p/2, 0, 1))
m <- rep(1, n)
y <- rbinom(n, m, as.numeric(1/(1 + exp(-x %*% as.matrix(beta)))))
```

```
partitions <- list(groups=rep(c(1, 2), each=p/2))

## calculate cross-validated predictions and performance measures
fit.cv.gren <- cv.gren(x, y, m, partitions=partitions, fix.lambda=TRUE)
```

dataCervical *Contains three R-objects, including the data and the binary response*

Description

The three objects are: `mirCerv`: 772 sequenced microRNAs for 56 samples; `respCerv`: binary response coded as healthy and pre-cursor lesions for cervical cancer; and `mirCons`: conservation status of the microRNAs in three levels coded as only found in humans, found in most mammals, and found in most vertebrates

Usage

```
data(dataCervical)
```

Format

The formats are:

mirCerv matrix [1:56, 1:772]

respCerv Factor w/ 2 levels "CIN3", "Normal"

mirCons Factor w/ 3 levels "NotCons", "Mammals", "Broadly"

Details

This data is used for illustration in the vignette of the [gren](#) package.

Source

Novianti, P.W., Snoek, B.C., Wilting, S.M., and van de Wiel, M.A. (2017). Better diagnostic signatures from RNAseq data through use of auxiliary co-data. *Bioinformatics*, 33, 1572–1574.

References

Münch, M.M., Peeters, C.F.W., van der Vaart, A.W., and van de Wiel, M.A. (2018). Adaptive group-regularized logistic elastic net regression. [arXiv:1805.00389v1 \[stat.ME\]](#).

Examples

```
data(dataCervical)
str(mirCerv)
str(respCerv)
str(mirCons)
```

`dataColon`*Contains four R-objects, including the data and the binary response*

Description

The four objects are: `mirCol`: 2114 sequenced microRNAs for 88 samples; `unpenCol`: 4 clinical covariates for 88 samples; `respCol`: binary response coded as progressive disease and benefitted from therapy; `mirExpr`: expression levels of microRNAs from previous study coded as not differentially expressed, medium significantly expressed, and highly significantly expressed.

Usage

```
data(dataColon)
```

Format

The formats are:

mirCol matrix [1:88, 1:2114]

unpenCol data.frame: 88 obs. of 4 variables

respCol Factor w/ 2 levels "Progr", "TherBenefit"

mirExpr Factor w/ 3 levels "nonExpr", "medExpr", "highExpr"

Details

This data is used for illustration in the vignette of the [gren](#) package and the corresponding paper (see references).

Source

Neerincx, M., Poel, D., Sie, D.L.S., van Grieken, N.C.T., Shankaraiah, R.C., van der Wolf - de Lijster, F.S.W., van Waesberghe, J.H.T.M., Burggraaf, J.D., Eijk, P.P., Verhoef, C., Ylstra, B., Meijer, G.A., van de Wiel, M.A., Buffart, T.E., and others. (2018). Combination of a six microRNA expression profile with four clinicopathological factors improves response prediction to systemic treatment in patients with advanced colorectal cancer. Submitted.

Neerincx, M., Sie, D.L.S., van de Wiel, M.A., van Grieken, N.C.T., Burggraaf, J.D., Dekker, H., Eijk, P.P., Ylstra, B., Verhoef, C., Meijer, G.A., Buffart, T.E., and others. (2015). MiR expression profiles of paired primary colorectal cancer and metastases by next-generation sequencing. *Oncogenesis*, 4, e170.

References

Münch, M.M., Peeters, C.F.W., van der Vaart, A.W., and van de Wiel, M.A. (2018). Adaptive group-regularized logistic elastic net regression. [arXiv:1805.00389v1 \[stat.ME\]](#).

Examples

```
data(dataColon)
str(mirCol)
str(unpenCol)
str(respCol)
str(mirExpr)
```

denet	<i>The elastic net prior distribution</i>
-------	---

Description

Density function and random number generator for the elastic net prior distribution.

Usage

```
denet(x, lambda1=1, lambda2=1, log=FALSE)

renet(n, lambda1=1, lambda2=1)
```

Arguments

x	vector of quantiles
n	number of samples
lambda1	lambda1 parameter value
lambda2	lambda2 parameter value
log	should the logarithm of the density be returned

Details

The elastic net prior density has density:

$$f(x) = g(\lambda_1, \lambda_2) e^{-0.5 * (\lambda_1 |x| + \lambda_2 x^2)}$$

Value

denet gives the density of the input x. renet gives a vector of length n of random values.

Author(s)

Magnus M. Münch <m.munch@vumc.nl>

References

Münch, M.M., Peeters, C.F.W., van der Vaart, A.W., and van de Wiel, M.A. (2018). Adaptive group-regularized logistic elastic net regression. arXiv:1805.00389v1 [stat.ME].

Examples

```
## Create data
n <- 100
x <- renet(n)
hist(x)

## Calculate density
dens <- denet(x)
plot(sort(x), dens[order(x)])
```

gren

Group-regularized logistic elastic net regression

Description

Function that estimates a group-regularized elastic net model.

Usage

```
gren(x, y, m=rep(1, nrow(x)), unpenalized=NULL, partitions=NULL, alpha=0.5,
     lambda=NULL, intercept=TRUE, monotone=NULL, psel=TRUE, compare=TRUE,
     posterior=FALSE, nfolds=nrow(x), foldid=NULL, trace=TRUE,
     init=list(lambdag=NULL, mu=NULL, sigma=NULL, chi=NULL, ci=NULL),
     control=list(epsilon=0.001, maxit=500, maxit.opt=1000, maxit.vb=100))
```

Arguments

x	feature data as either numeric matrix or data.frame of numeric variables.
y	response as either a numeric with binomial/binary successes of length nrow(x) or a matrix of nrow(x) rows and two columns, where the first column contains the binomial/binary failures and the second column the binomial/binary successes.
m	numeric of length nrow(x) that contains the number of Bernoulli trials.
unpenalized	Optional numeric matrix or data.frame of numeric unpenalized covariates of nrow(x) rows.
partitions	list that contains the (possibly multiple) partitions of the data. Every list object corresponds to one partition, where every partition is a numeric of length ncol(x) containing the group ids of the features.
alpha	proportion of L1 penalty as a numeric of length 1.
lambda	global penalty parameter. The default NULL will result in estimation by cross-validation.
intercept	logical to indicate whether an intercept should be included.
monotone	list of two logical vectors of length length(partitions). The first one monotone indicates whether the corresponding partition's penalty parameters should be monotonically estimates, the second vector decreasing indicates whether the monotone penalty parameters are decreasing with group number.

<code>pselect</code>	either a numeric vector that indicates the number of features to select or a logical. If TRUE feature selection is done by letting <code>glmnet</code> determine the penalty parameter sequence.
<code>compare</code>	logical, if TRUE, a regular non-group-regularized model is estimated.
<code>posterior</code>	if TRUE, the full variational Bayes posterior is returned.
<code>nfolds</code>	numeric of length 1 with the number of folds used in the cross-validation of the global lambda. The default is <code>nrow(x)</code> .
<code>foldid</code>	optional numeric vector of length <code>nrow(x)</code> with the fold assignments of the observations.
<code>trace</code>	if TRUE, progress of the algorithm is printed.
<code>init</code>	optional list containing the starting values of the iterative algorithm. See Details for more information.
<code>control</code>	a list of algorithm control parameters. See Details for more information.

Details

This is the main function of the package that estimates a group-regularized elastic net regression. The elastic net penalty's proportion of L1-norm penalisation is determined by `alpha`. `alpha` close to 0 implies more ridge-like penalty, while `alpha` close to 1 implies lasso-like penalty. The algorithm is a two-step procedure: first, a global lambda penalty is estimated by cross-validation. Next, the groupwise lambda multipliers are estimated by an EM algorithm. The EM algorithm consists of: i) an expectation step in which the expected marginal likelihood of the penalty multipliers is iteratively approximated by a variational Bayes EM algorithm and ii) a maximisation step in which the approximate expected marginal likelihood is maximised with respect to the penalty multipliers. After convergence of the algorithm an (optional) frequentist elastic net model is fit using the estimated penalty multipliers by setting `pselect=TRUE` or by setting `pselect` to a numeric vector.

The user may speed up the procedure by specifying initial values for the EM algorithm in `init`. `init` is a list that contains:

`lambda0` initial values for λ_g in a list of length `length(partitions)`.

`mu` initial values for the μ_j in a numeric vector of length `ncol(x) + ncol(unpenalized) + intercept`.

`chi` initial values for the χ_j in a numeric vector of length `ncol(x)`.

`ci` initial values for the c_i in a numeric vector of length `nrow(x)`.

`sigma` The initial values for the Σ_{ij} in a matrix of numerics with `ncol(x)` rows and columns.

`control` is a list with parameters to control the estimation procedure. It consists of the following components:

`epsilon` numeric with the relative convergence tolerance. Default is `epsilon=0.001`.

`maxit` numeric with whole number that gives the maximum number of iterations to update the `lambda0`. Default is `maxit=500`.

`maxit.opt` numeric with whole number that gives the maximum number of iterations to numerically maximise the `lambda0`. Maximisation occurs at every iteration. Default is `maxit.opt=1000`.

`maxit.vb` numeric with whole number that gives the maximum number of iterations to update the variational parameters `mu`, `sigma`, `chi`, and `ci`. One full update sequence per iteration. Default is `maxit=100`.

Value

Function returns an S3 list object of class `gren` containing output with the following components:

<code>call</code>	The function call that produced the output.
<code>alpha</code>	proportion of L1 penalty as a numeric of length 1.
<code>lambda</code>	global penalty parameter as numeric. Estimated by cross-validation if <code>lambda=NULL</code> .
<code>lambda.seq</code>	list with full sequence of penalty multipliers over iterations.
<code>lambda</code>	list with final estimates of penalty multipliers.
<code>vb.post</code>	list with variational posterior parameters μ_j , σ_{ij} , c_i , and chi_j .
<code>freq.model</code>	frequentist elastic net model as output of <code>glmnet</code> call. NULL if <code>pse1=FALSE</code> .
<code>iter</code>	list with number of iterations of <code>lambda</code> estimation, with number of optimisation iterations of <code>lambda</code> , and number of variational Bayes iterations.
<code>conv</code>	list of logicals with convergence of <code>lambda</code> sequence, optimisation steps, and variational Bayes iterations.
<code>args</code>	list with input arguments of <code>gren</code> call.

Author(s)

Magnus M. Münch <m.munch@vumc.nl>

References

Münch, M.M., Peeters, C.F.W., van der Vaart, A.W., and van de Wiel, M.A. (2018). Adaptive group-regularized logistic elastic net regression. arXiv:1805.00389v1 [stat.ME].

See Also

[predict.gren](#), [coef.gren](#), [cv.gren](#)

Examples

```
## Create data
p <- 1000
n <- 100
set.seed(2018)
x <- matrix(rnorm(n*p), ncol=p, nrow=n)
beta <- c(rnorm(p/2, 0, 0.1), rnorm(p/2, 0, 1))
m <- rep(1, n)
y <- rbinom(n, m, as.numeric(1/(1 + exp(-x %*% as.matrix(beta)))))
partitions <- list(groups=rep(c(1, 2), each=p/2))

## estimate model
fit.gren <- gren(x, y, m, partitions=partitions)
```

predict.gren	<i>Predictions and coefficients from gren model.</i>
--------------	--

Description

Create predictions from new data using a fitted gren model/retrieve coefficients from fitted model. Both are S3 methods.

Usage

```
## S3 method for class 'gren'  
predict(object, newx, unpenalized=NULL, s=NULL,  
        type=c("groupreg", "regular"), ...)  
  
## S3 method for class 'gren'  
coef(object, s=NULL, type=c("groupreg", "regular"),...)
```

Arguments

object	A fitted gren model.
newx	New data for which to do predictions.
unpenalized	New unpenalized data for which to do predictions.
s	Value of lambda for which to create predictions/coefficients, may be a vector.
type	Either groupreg, which creates predictions/coefficients of group-regularized model, or regular, for predictions/coefficients from regular model.
...	Further arguments to be passed.

Details

This are the predict/coefficient functions of the gren package.

Value

predict returns a numeric matrix with predicted probabilities. coef returns a matrix with coefficients.

Author(s)

Magnus M. Münch <m.munch@vumc.nl>

References

Münch, M.M., Peeters, C.F.W., van der Vaart, A.W., and van de Wiel, M.A. (2018). Adaptive group-regularized logistic elastic net regression. arXiv:1805.00389v1 [stat.ME].

See Also[gren,coef.gren](#)**Examples**

```
## Create data
p <- 1000
n <- 100
set.seed(2018)
x <- matrix(rnorm(n*p), ncol=p, nrow=n)
beta <- c(rnorm(p/2, 0, 0.1), rnorm(p/2, 0, 1))
m <- rep(1, n)
y <- rbinom(n, m, as.numeric(1/(1 + exp(-x %*% as.matrix(beta)))))
partitions <- list(groups=rep(c(1, 2), each=p/2))

## estimate model
fit.gren <- gren(x, y, m, partitions=partitions)

## create new data
xnew <- matrix(rnorm(n*p), ncol=p, nrow=n)

## create predictions/coefficients
preds <- predict(fit.gren, xnew, type="groupreg")
coefs <- coef(fit.gren, type="groupreg")
```

Index

* datasets

dataCervical, 5

dataColon, 6

* package

gren-package, 2

coef.gren, 2, 4, 10, 12

coef.gren (predict.gren), 11

cv.gren, 2, 3, 10

dataCervical, 5

dataColon, 6

denet, 2, 7

glmnet, 9

gren, 2–6, 8, 12

gren-package, 2

mirCerv (dataCervical), 5

mirCol (dataColon), 6

mirCons (dataCervical), 5

mirExpr (dataColon), 6

predict.gren, 2, 4, 10, 11

renet, 2

renet (denet), 7

respCerv (dataCervical), 5

respCol (dataColon), 6

unpenCol (dataColon), 6