Package ‘BootValidation’

August 14, 2018

Type Package

Title Adjusting for Optimism in 'glmnet' Regression using Bootstrapping

Version 0.1.65

Author Antonio Jose Canada Martinez

Maintainer Antonio Jose Canada Martinez <ancamar2@gmail.com>

Description

Main objective of a predictive model is to provide accurated predictions of a new observations. Unfortunately we don't know how well the model performs. In addition, at the current era of omic data where p >> n, is not reasonable applying internal validation using data-splitting. Under this background a good method to assessing model performance is applying internal bootstrap validation
(Harrell Jr, Frank E (2015) <doi:10.1007/978-1-4757-3462-1>.) This package provides bootstrap validation
for the linear, logistic, multinomial and cox 'glmnet' models as well as lm and glm models.

License GPL (>= 2)

Encoding UTF-8

LazyData true

Imports glmnet, pbapply, pROC, parallel, survival, risksetROC

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-08-14 13:00:07 UTC

R topics documented:

plot.bootVal *Plot for repeat_cv results*

Description

Plots a grid of slices from the estimates of the repeat_cv function

Usage

```
## S3 method for class 'bootVal'
plot(x, order = TRUE,
  n = length(unique(unlist(x$varImportance))), ...)
```

Arguments

`x`	A vboot object
`order`	order plot by importance
`n`	Variables to be displayed
`...`	further arguments passed to plot

Value

validation

print.bootVal *Print function*

Description

internal function to print vboot object

Usage

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

Arguments

`x`	A vboot object
`...`	Further arguments of generic function

Value

validation

vboot *Generic function for bootstrap validation*

Description

Validate 'glmnet' linear, logistic or cox regression using bootstrap.

Usage

```
vboot(fit, x, y, s, nfolds = 5, B = 200, cv_replicates = 100,
  lambda = TRUE, n_cores = max(1, parallel::detectCores() - 1))
```

Arguments

`fit`	Object from glmnet fit.
`x`	A matrix of the predictors, each row is an observation vector.
`y`	A vector of response variable. It should be quantitative for lineal regression, a factor with two levels for logistic regression, a factor with more than two levels for multinomial regression or a two-column matrix with columns named 'time' and 'status' for cox regression.
`s`	Value of the penalty parameter "lambda" selected from the original 'cv.glmnet'.
`nfolds`	Number of folds for cross validation as in 'cv.glmnet'.
`B`	Number of bootsrap samples.
`cv_replicates`	Number of replicates for the cross-validation step.
`lambda`	By default, the validation is adjusted using 'lambda.1se' which has error within 1 standard error of the best model. If 'FALSE' the 'lambda.min' referered to the lowest CV error will be used.
`n_cores`	number of cores to use in parallel. Default detectCores()-1.

References

Jerome Friedman, Trevor Hastie, Robert Tibshirani (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software, 33(1), 1-22. URL http://www.jstatsoft.org/v33/i01/.

Noah Simon, Jerome Friedman, Trevor Hastie, Rob Tibshirani (2011). Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent. Journal of Statistical Software, 39(5), 1-13. URL http://www.jstatsoft.org/v39/i05/.

Harrell Jr, F. E. (2015). Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis. Springer.

Gordon C.S. Smith, Shaun R. Seaman, Angela M. Wood, Patrick Royston, Ian R. White (2014). Correcting for Optimistic Prediction in Small Data Sets, American Journal of Epidemiology, Volume 180, Issue 3, 1 August 2014, Pages 318-324, https://doi.org/10.1093/aje/kwu140

Examples

```
# Create the data
set.seed(25)
x <- matrix(rnorm(80),ncol=4)
y <- x[,4]*0.8+x[,3]*0.4+rnorm(20)
# Fit glmnet model
fit_enet <- glmnet::glmnet(x, y, alpha = 0.5)
# Bootstrap validation
vboot(fit_enet, x, y, nfolds = 3, B = 2, s = 0.5, cv_replicates = 5, n_cores = 1)
```

`vboot.coxnet` *Internal bootstraping validation cox glmnet model*

Description

Validate glmnet cox regression using bootstrap.

Usage

```
## S3 method for class 'coxnet'
vboot(fit, x, y, s, nfolds = 5, B = 200,
  cv_replicates = 100, lambda = TRUE, n_cores = max(1,
  parallel::detectCores() - 1))
```

Arguments

`fit` Object from glmnet fit.

`x` A matrix of the predictors, each row is an observation vector.

`y` Should be a two-column matrix with columns named ’time’ and ’status’ as in ’glmnet’.

`s` Value of the penalty parameter "lambda" selected from the original ’cv.glmnet’.

`nfolds` Number of folds for cross validation as in ’cv.glmnet’.

`B` Number of bootsrap samples.

`cv_replicates` Number of replicates for the cross-validation step.

`lambda` By default, the validation is adjusted using ’lambda.1se’ which has error within 1 standard error of the best model. If ’FALSE’ the ’lambda.min’ referered to the lowest CV error will be used.

`n_cores` number of cores to use in parallel. Default detectCores()-1.

vboot.elnet *Internal bootstraping validation linear glmnet model*

Description

Validate glmnet linear regression using bootstrap.

Usage

```
## S3 method for class 'elnet'
vboot(fit, x, y, s, nfolds = 5, B = 200,
  cv_replicates = 100, lambda = TRUE, n_cores = max(1,
  parallel::detectCores() - 1))
```

Arguments

`fit`	Object from glmnet fit.
`x`	A matrix of the predictors, each row is an observation vector.
`y`	A vector of response variable. Should be numeric.
`s`	Value of the penalty parameter "lambda" selected from the original 'cv.glmnet'.
`nfolds`	Number of folds for cross validation as in 'cv.glmnet'.
`B`	Number of bootsrap samples.
`cv_replicates`	Number of replicates for the cross-validation step.
`lambda`	By default, the validation is adjusted using 'lambda.1se' which has error within 1 standard error of the best model. If 'FALSE' the 'lambda.min' referered to the lowest CV error will be used.
`n_cores`	number of cores to use in parallel. Default detectCores()-1.

vboot.glm *Internal bootstraping validation logistic model*

Description

Validate logistic regression using bootstrap.

Usage

```
## S3 method for class 'glm'
vboot(fit, x = NULL, y = NULL, s = NULL, nfolds = NULL,
  B = 200, cv_replicates = NULL, lambda = NULL, n_cores = max(1,
  parallel::detectCores() - 1))
```

Arguments

`fit` Object from glm fit

`x` A matrix of the predictors, each row is an observation vector.

`y` A vector of response variable. It should be quantitative for lineal regression, a factor with two levels for logistic regression or a two-column matrix with columns named 'time' and 'status' for cox regression.

`s` Value of the penalty parameter "lambda" selected from the original 'cv.glmnet'.

`nfolds` Number of folds for cross validation as in 'cv.glmnet'.

`B` Number of bootsrap samples

`cv_replicates` Number of replicates for the cross-validation step

`lambda` By default, the validation is adjusted using 'lambda.1se' which has error within 1 standard error of the best model. If 'FALSE' the 'lambda.min' refereed to the lowest CV error will be used.

`n_cores` number of cores to use in parallel. Default detectCores()-1

`vboot.lm` *Internal bootstraping validation linear model*

Description

Validate linear regression using bootstrap.

Usage

```
## S3 method for class 'lm'
vboot(fit, x = NULL, y = NULL, s = NULL, nfolds = NULL,
  B = 200, cv_replicates = NULL, lambda = NULL, n_cores = max(1,
  parallel::detectCores() - 1))
```

Arguments

`fit` Object from lm fit

`x` A matrix of the predictors, each row is an observation vector.

`y` A vector of response variable. It should be quantitative for lineal regression, a factor with two levels for logistic regression or a two-column matrix with columns named 'time' and 'status' for cox regression.

`s` Value of the penalty parameter "lambda" selected from the original 'cv.glmnet'.

`nfolds` Number of folds for cross validation as in 'cv.glmnet'.

`B` Number of bootsrap samples

`cv_replicates` Number of replicates for the cross-validation step

`lambda` By default, the validation is adjusted using 'lambda.1se' which has error within 1 standard error of the best model. If 'FALSE' the 'lambda.min' refereed to the lowest CV error will be used.

`n_cores` number of cores to use in parallel. Default detectCores()-1

vboot.lognet *Internal bootstraping validation logistic glmnet model*

Description

Validate glmnet logistic regression using bootstrap.

Usage

```
## S3 method for class 'lognet'
vboot(fit, x, y, s, nfolds = 5, B = 200,
  cv_replicates = 100, lambda = TRUE, n_cores = max(1,
  parallel::detectCores() - 1))
```

Arguments

`fit` Object from glmnet fit.

`x` A matrix of the predictors, each row is an observation vector..

`y` A vector of response variable. Should be a factor with two levels.

`s` Value of the penalty parameter "lambda" selected from the original 'cv.glmnet'.

`nfolds` Number of folds for cross validation as in cv.glmnet.

`B` Number of bootsrap samples.

`cv_replicates` Number of replicates for the cross-validation step in 'cv.glmnet'.

`lambda` By default, the validation is adjusted using 'lambda.1se' which has error within 1 standard error of the best model. If 'FALSE' the 'lambda.min' referered to the lowest CV error will be used.

`n_cores` number of cores to use in parallel. Default detectCores()-1.

vboot.multnet *Internal bootstraping validation multinomial glmnet model*

Description

Validate glmnet logistic regression using bootstrap.

Usage

```
## S3 method for class 'multnet'
vboot(fit, x, y, s, nfolds = 5, B = 200,
  cv_replicates = 100, lambda = TRUE, n_cores = max(1,
  parallel::detectCores() - 1))
```

Arguments

`fit` Object from glmnet fit.

`x` A matrix of the predictors, each row is an observation vector.

`y` A vector of response variable. Should be a factor with two levels.

`s` Value of the penalty parameter "lambda" selected from the original ’cv.glmnet’.

`nfolds` Number of folds for cross validation as in cv.glmnet.

`B` Number of bootsrap samples.

`cv_replicates` Number of replicates for the cross-validation step in ’cv.glmnet’.

`lambda` By default, the validation is adjusted using ’lambda.1se’ which has error within 1 standard error of the best model. If ’FALSE’ the ’lambda.min’ referered to the lowest CV error will be used.

`n_cores` number of cores to use in parallel. Default detectCores()-1.

Index