Package 'boss'

October 15, 2019

Type Package

Title Best orthogonalized subset selection (BOSS)

Version 0.1.0

Date 2019-11-01

Maintainer Sen Tian <stian@stern.nyu.edu>

Description An implementation of best orthogonalized subset selection (BOSS) and forward stepwise selection (fs), together with feasible selection rules to choose the optimal candidate subset.

License GPL (>=2) | file LICENSE

Encoding UTF-8

LazyData true

Imports Matrix,
Rcpp,
RcppArmadillo,
stats,
MASS

RoxygenNote 6.1.1

Suggests knitr,
rmarkdown

VignetteBuilder knitr

LinkingTo Rcpp,
RcppArmadillo

URL http://github.com/sentian/boss

BugReports http://github.com/sentian/boss/issues

R topics documented:

boss *Best orthogonalized subset selection (BOSS).*

Description

- Compute the solution path of BOSS and forward stepwise selection (FS).
- Compute various information criteria based on a heuristic degrees of freedom that can serve as the selection rule to choose the optimal subset given by BOSS. Only work when n>p.

Usage

```
boss(x, y, intercept = TRUE, fs.only = FALSE, hdf.ic.boss = TRUE,
  mu = NULL, sigma = NULL, ...)
```

Arguments

x	A matrix of predictors, with nrow(x)=length(y)=n observations and ncol(x)=p predictors. Intercept shall not be included.
y	A vector of response variable, with length(y)=n.
intercept	Whether to include an intercept term.
fs.only	Whether to ignore BOSS and perform FS only.
hdf.ic.boss	Whether to calculate the heuristic degrees of freedom (hdf) and information criteria (IC) for BOSS. IC includes AIC, BIC, AICc, BICc, GCV, Cp. Note that if the option fs.only=TRUE or n<=p, hdf.ic.boss=FALSE no matter what.
mu	True mean vector, used in the calculation of hdf. Default is NULL, and is estimated via full OLS.
sigma	True standard deviation of the error, used in the calculation of hdf. Default is NULL, and is estimated via full OLS.
...	Extra parameters to allow flexibility. Currently none argument allows or requires, just for the convinience of call from other parent functions like cv.boss.

Details

This function computes the full solution path given by FS and (or) BOSS on a given dataset (x,y) with n observations and p predictors. Meanwhile, in the case where n>p, it calculates the heuristic degrees of freedom for BOSS, and various information criteria, which can further be used to select the optimal candidate along the path. Please refer to the example section below for implementation details and Tian et al. (2018) for methodology details.

Value

- beta_fs: A matrix of regression coefficients for each step performed by FS, from a null model until stop, with nrow=p and ncol=min(n,p)+1, where min(n,p) is the maximum number of steps performed.
- beta_boss: A matrix of regression coefficients for each step performed by BOSS, with nrow=p and ncol=min(n,p)+1. Note that unlike beta_fs and due to the nature of BOSS, the number of non-zero components in columns of beta_boss may not be unique, i.e. there maybe multiple columns corresponding to the same size of subset. beta_boss=NULL if the option fs.only=TRUE.

- steps_fs: A vector of numbers representing which predictor joins at each step, with `length(steps_fs)=min(n,p)`.
- hdf_boss: A vector of heuristic degrees of freedom (hdf) for BOSS, with `length(hdf_boss)=p+1`. Note that `hdf_boss=NULL` if `n<=p` or `hdf.ic.boss=FALSE`.
- IC_boss: A list of information criteria (IC) for BOSS, where each element in the list is a vector representing values of a given IC for each candidate subset of BOSS (or each column in beta_boss). The output IC includes AIC, BIC, AICc, BICc, GCV and Mallows' Cp. Note that each IC is calculated by plugging in hdf_boss.

Author(s)

Sen Tian

References

Tian, Hurvich and Simonoff (2019), On the use of information criterion in least squares based subset selection problems. (Link to be added)

See Also

`predict` and `coef` methods for "boss" object, and the `cv.boss` function

Examples

```
## Generate a trivial dataset, X has mean 0 and norm 1, y has mean 0
set.seed(11)
n = 20
p = 5
x = matrix(rnorm(n*p), nrow=n, ncol=p)
x = scale(x, center = colMeans(x))
x = scale(x, scale = sqrt(colSums(x^2)))
beta = c(1, 1, 0, 0, 0)
y = x%*%beta + scale(rnorm(n, sd=0.01), center = TRUE, scale = FALSE)

## Fit the model
boss_result = boss(x, y)

## Get the coefficient vector selected by AICc-hdf (S3 method for boss)
beta_boss_aicc = coef(boss_result)$boss
# the above is equivalent to the following
beta_boss_aicc = boss_result$beta_boss[, which.min(boss_result$IC_boss$aicc), drop=FALSE]
## Get the fitted values of BOSS-AICc-hdf (S3 method for boss)
mu_boss_aicc = predict(boss_result, newx=x)$boss
# the above is equivalent to the following
mu_boss_aicc = cbind(1,x) %*% beta_boss_aicc

## Repeat the above process, but using Cp-hdf instead of AICc-hdf
## coefficient vector
beta_boss_cp = coef(boss_result, method.boss='cp')$boss
beta_boss_cp = boss_result$beta_boss[, which.min(boss_result$IC_boss$cp), drop=FALSE]
## fitted values of BOSS-Cp-hdf
mu_boss_cp = predict(boss_result, newx=x, method.boss='cp')$boss
mu_boss_cp = cbind(1,x) %*% beta_boss_cp
```

`calc.ic` *Calculate information criterion.*

Description

Calculate a specified information criterion (IC) for an estimate or a group of estimates. Such IC includes AIC, BIC, AICc, BICc, GCV and Mallows' Cp.

Usage

```
calc.ic(y_hat, y, method = c("aicc", "bicc", "aic", "bic", "gcv", "cp"),
  df, sigma = NULL)
```

Arguments

`y_hat`	A vector of fitted values with `length(y_hat)=length(y)=n`, or a matrix, with `nrow(coef)=length(y)=n` and `ncol(y_hat)=m`, containing m different fits.
`y`	A vector of response variable, with `length(y)=n`.
`method`	A specified IC to calculate. Default is AICc ('aicc'). Other choices include AIC ('aic'), BIC ('bic'), BICc ('bicc'), GCV ('gcv') and Mallows' Cp ('cp').
`df`	A number if y_hat is a vector, or a vector with `length(df)=ncol(y_hat)=m` if y_hat is a matrix. df represents the degrees of freedom for each fit.
`sigma`	Standard deviation of the error term. It only needs to be specified if `method='cp'`.

Details

This function enables the computation of various common IC for model fits, which can further be used to choose the optimal fit. This allows user comparing the effect of different IC. In order to calculate an IC, degrees of freedoms (df) needs to be specified. To be more specific, here are the formulas used to calculate each IC:

$$AIC = \log(\frac{RSS}{n}) + 2\frac{df}{n}$$

$$BIC = \log(\frac{RSS}{n}) + \log(n)\frac{df}{n}$$

$$AICc = \log(\frac{RSS}{n}) + 2\frac{df+1}{n-df-2}$$

$$BICc = \log(\frac{RSS}{n}) + \log(n)\frac{df+1}{n-df-2}$$

$$GCV = \frac{RSS}{(n-df)^2}$$

$$Mallows'Cp = RSS + 2 \times \sigma^2 \times df$$

Value

The value(s) of the specified IC for each fit.

Author(s)

Sen Tian

Examples

```
## Generate a trivial dataset, X has mean 0 and norm 1, y has mean 0
set.seed(11)
n = 20
p = 5
x = matrix(rnorm(n*p), nrow=n, ncol=p)
x = scale(x, center = colMeans(x))
x = scale(x, scale = sqrt(colSums(x^2)))
beta = c(1, 1, 0, 0, 0)
y = x%*%beta + scale(rnorm(20, sd=0.01), center = TRUE, scale = FALSE)

## Fit the model
boss_result = boss(x, y)
## Print the values of AICc-hdf for all subsets given by BOSS
print(boss_result$IC_boss$aicc)
## calculate them manually using the calc.ic function
y_hat = cbind(rep(1,n),x)%*%boss_result$beta_boss
print(calc.ic(y_hat, y, df=boss_result$hdf_boss))
```

---

coef.boss *Select coefficient vector(s) from boss object.*

Description

This function returns coefficient vector(s) for given step(s) of FS and BOSS. For BOSS, it can also return the optimal coefficient vector selected by AICc (by default) or other IC.

Usage

```
## S3 method for class 'boss'
coef(object, select.fs = NULL, select.boss = NULL,
  method.boss = c("aicc", "bicc", "aic", "bic", "gcv", "cp"), ...)
```

Arguments

`object`	The boss object, returned from calling 'boss' function.
`select.fs`	Given step(s) of FS, corresponding to columns in the coefficient matrix. For example, the first column in beta_fs corresponds to step 1, the second column corresponds to step 2, etc. We enforce the first column, that has all 0 entries, to be step 1, just for convinicence of usage.
`select.boss`	Given columns(s) in beta_boss, similar to select.fs.
`method.boss`	Which IC is used to select the optimal coefficient vector for BOSS.
`...`	Extra arguments (unused for now)

Details

If `select.fs` or `select.boss` is specified, the function returns corresponding column(s) in the coefficient matrix.

If `select.fs` is unspecified, we return the entire coefficient matrix (`beta_fs`) for FS, since we currently do not have a hands-on way of calculating IC for FS. But user can use their own rules and specify e.g. `select.fs=which.min(rule)` in order to pick the optimal coefficient vector.

If `select.boss` is unspecified, we will try to return the optimal coefficient vector selected by AICc-hdf (other choice of IC can be specified in `method.boss`). The only exception is when n>=p, where hdf is not well defined, and we will return the entire coefficient matrix.

Value

- fs: The chosen coefficient vector(s) for FS.
- boss: The chosen coefficient vector(s) for FS.

Examples

```
# See the example in the section of \code{boss}. Or type ?boss in R.
```

`coef.cv.boss` *Select coefficient vector based on cross validation (CV).*

Description

This function returns coefficient vector that minimizes out-of-sample (OOS) cross validation score.

Usage

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

Arguments

`object`	The cv.boss object, returned from calling 'cv.boss' function.
`...`	Extra arguments (unused for now).

Value

- fs: The chosen coefficient vector for FS.
- boss: The chosen coefficient vector for FS.

Examples

```
# See the example in the section of \code{cv.boss}. Or type ?cv.boss in R.
```

`cv.boss` *Cross validation for BOSS.*

Description

Cross validation for BOSS and FS.

Usage

```
cv.boss(x, y, n.folds=10, n.rep=1, ...)
```

Arguments

`x`	A matrix of predictors, see `boss`.
`y`	A vector of response variable, see `boss`.
`n.folds`	The number of cross validation folds.
`n.rep`	The number of replications of cross validation.
`...`	Arguments to `boss`.

Details

This function fits BOSS and FS (`boss`) on the full dataset, and performs `n.folds` cross validation. The cross validation process can be repeated `n.rep` times to evaluate the out-of-sample (OOS) performance for the candidate subsets given by both methods.

Value

- boss: An object `boss` that fits on the full dataset.
- n.folds: The number of cross validation folds.
- cvm.fs: Mean OOS deviance for each candidate given by FS.
- cvm.boss: Mean OSS deviance for each candidate given by BOSS.
- i.min.fs: The index of minimum cvm.fs.
- i.min.boss: The index of minimum cvm.boss.

Author(s)

Sen Tian

References

Tian, Hurvich and Simonoff (2019), On the use of information criterion in least squares based subset selection problems. (Link to be added)

See Also

`predict` and `coef` methods for "cv.boss" object, and the `boss` function

Examples

```
## Generate a trivial dataset, X has mean 0 and norm 1, y has mean 0
set.seed(11)
n = 20
p = 5
x = matrix(rnorm(n*p), nrow=n, ncol=p)
x = scale(x, center = colMeans(x))
x = scale(x, scale = sqrt(colSums(x^2)))
beta = c(1, 1, 0, 0, 0)
y = x%*%beta + scale(rnorm(20, sd=0.01), center = TRUE, scale = FALSE)

## Perform 10-fold CV without replication
boss_cv_result = cv.boss(x, y)
## Get the coefficient vector of BOSS that gives minimum CV OSS score (S3 method for cv.boss)
beta_boss_cv = coef(boss_cv_result)$boss
# the above is equivalent to
boss_result = boss_cv_result$boss
beta_boss_cv = boss_result$beta_boss[, boss_cv_result$i.min.boss, drop=FALSE]
## Get the fitted values of BOSS-CV (S3 method for cv.boss)
mu_boss_cv = predict(boss_cv_result, newx=x)$boss
# the above is equivalent to
mu_boss_cv = cbind(1,x) %*% beta_boss_cv

## Get the coefficient vector of FS that gives minimum CV OSS score (S3 method for cv.boss)
beta_fs_cv = coef(boss_cv_result)$fs
## Get the fitted values of FS-CV (S3 method for cv.boss)
mu_fs_cv = predict(boss_cv_result, newx=x)$fs
```

`predict.boss` *Prediction given new data entries.*

Description

This function returns the prediction(s) given new observation(s), for FS and BOSS, where the optimal coefficient vector is chosen via certain selection rule.

Usage

```
## S3 method for class 'boss'
predict(object, newx, ...)
```

Arguments

`object`	The boss object, returned from calling 'boss' function.
`newx`	A new data entry or several entries. It can be a vector, or a matrix with `nrow(newx)` being the number of new entries and `ncol(newx)=p` being the number of predictors. The function takes care of the intercept, NO need to add 1 to `newx`.
`...`	Extra arguments to be plugged into `coef`, such as `select.boss`, see the description of `coef` for more details.

Details

The function basically calculates $x * coef$, where `coef` is a coefficient vector chosen by a selection rule. See more details about the default and available choices of the selection rule in the description of `coef.boss`.

Value

- fs: The prediction(s) for FS.
- boss: The prediction(s) for BOSS.

Examples

```
#See the example in the section of \code{boss}. Or type ?boss in R.
```

`predict.cv.boss` *Prediction given new data entries.*

Description

This function returns the prediction(s) given new observation(s), for FS and BOSS, where the optimal coefficient vector is chosen via cross-validation.

Usage

```
## S3 method for class 'cv.boss'
predict(object, newx, ...)
```

Arguments

`object`	The cv.boss object, returned from calling 'cv.boss' function.
`newx`	A new data entry or several entries. It can be a vector, or a matrix with `nrow(newx)` being the number of new entries and `ncol(newx)=p` being the number of predictors. The function takes care of the intercept, NO need to add `1` to `newx`.
`...`	Extra arguments (unused for now).

Value

- fs: The prediction for FS.
- boss: The prediction for BOSS.

Examples

```
# See the example in the section of \code{cv.boss}. Or type ?cv.boss in R.
```

Index