Package 'robustsubsets'

November 15, 2020

Type Package

Title Robust subset selection in linear regression

Version 1.0.2

Author Ryan Thompson

Maintainer Ryan Thompson <ryan.thompson@monash.edu>

Description Provides functionality for robust subset selection in linear regression.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports stats,
foreach,
doParallel,
abind,
gurobi,
Matrix,
ggplot2,
Rcpp

LinkingTo Rcpp,
RcppArmadillo

R topics documented:

`bss` *Best subset selection*

Description

Fits a sequence of best subset selection models. This function is just a wrapper for the `rss` function. The function solves the robust subset selection problem with `h=n`, using nonrobust measures of location and scale to standardise, as well as a nonrobust measure of prediction error in cross-validation.

Usage

```
bss(X, y, k = 0:min(nrow(X) - int, ncol(X), 20), int = T, mio = "min", ...)
```

Arguments

`X`	a matrix of predictors
`y`	a vector of the response
`k`	the number of predictors to minimise sum of squares over; by default a sequence from 0 to 20
`int`	a logical indicating whether to include an intercept
`mio`	one of 'min', 'all', or 'none' indicating whether to run the mixed-integer solver on the `k` that minimises the cv error, all `k`, or none at all
`...`	any other arguments

Value

See documentation for the `rss` function.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

Examples

```
# Generate training data
set.seed(1)
n <- 100
p <- 10
p0 <- 5
beta <- c(rep(1, p0), rep(0, p - p0))
X <- matrix(rnorm(n * p), n, p)
e <- rnorm(n)
```

```
y <- X %*% beta + e

# Fit best subset selection models
# Run the mixed-integer solver on the k that minimises the cv error
fit <- bss(X, y, k = 0:10, mio = 'min', n.core = 1)

# Extract model coefficients and generate predictions
coef(fit)
predict(fit, X)

# Plot coefficient profiles and cross-validation results
plot(fit, type = 'profile')
plot(fit, type = 'cv')
```

`coef.rss` *Coefficient function for rss object*

Description

Extracts coefficients for a given parameter pair `(k,h)`.

Usage

```
## S3 method for class 'rss'
coef(object, k = "k.min", h = "h.min", ...)
```

Arguments

`object`	an object of class `rss`
`k`	the number of predictors indexing the desired fit; 'k.min' uses best `k` from cross-validation
`h`	the number of observations indexing the desired fit; 'h.min' uses best `h` from cross-validation
`...`	any other arguments

Value

An array of coefficients.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

coef.rss.fit *Coefficient function for rss.fit object*

Description

Extracts coefficients for a given parameter pair (k,h).

Usage

```
## S3 method for class 'rss.fit'
coef(object, k = NULL, h = NULL, ...)
```

Arguments

object	an object of class rss.fit
k	the number of predictors indexing the desired fit
h	the number of observations indexing the desired fit
...	any other arguments

Value

An array of coefficients.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

plot.rss *Plot function for rss object*

Description

Plot the cross-validation results or coefficient profiles from robust subset selection.

Usage

```
## S3 method for class 'rss'
plot(x, type = "cv", ...)
```

Arguments

x	an object of class rss
type	one of 'cv' or 'profile'
...	any other arguments

Value

A plot of the cross-validation results or coefficient profiles.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

---

plot.rss.cv *Plot function for rss.cv object*

Description

Plot the cross-validation results from robust subset selection.

Usage

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

Arguments

x	an object of class rss.cv
...	any other arguments

Value

A plot of the cross-validation results.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

---

plot.rss.fit *Plot function for rss.fit object*

Description

Plot the coefficient profiles from robust subset selection.

Usage

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

Arguments

x	an object of class rss.fit
...	any other arguments

Value

A plot of the coefficient profiles.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

---

predict.rss *Predict function for rss object*

Description

Generate predictions given new data using a given parameter pair (k,h).

Usage

```
## S3 method for class 'rss'
predict(object, X.new, k = "k.min", h = "h.min", ...)
```

Arguments

object	an object of class rss
X.new	a matrix of new values for the predictors
k	the number of predictors indexing the desired fit; 'k.min' uses best k from cross-validation
h	the number of observations indexing the desired fit; 'h.min' uses best h from cross-validation
...	any other arguments

Value

An array of predictions.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

predict.rss.fit *Predict function for rss.fit object*

Description

Generate predictions for new data using a given parameter pair (k,h).

Usage

```
## S3 method for class 'rss.fit'
predict(object, X.new, k = NULL, h = NULL, ...)
```

Arguments

object	an object of class rss.fit
X.new	a matrix of new values for the predictors
k	the number of predictors indexing the desired fit
h	the number of observations indexing the desired fit
...	any other arguments

Value

An array of predictions.

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

rss *Robust subset selection*

Description

Fits a sequence of robust subset selection models and cross-validates the prediction error from these models.

Usage

```
rss(
  X,
  y,
  k = 0:min(nrow(X) - int, ncol(X), 20),
  h = function(n) round(seq(0.75, 1, 0.05) * n),
  int = T,
  mio = "min",
  ...
)
```

Arguments

X	a matrix of predictors
y	a vector of the response
k	the number of predictors to minimise sum of squares over; by default a sequence from 0 to 20
h	a function that takes the sample size that returns the number of observations to minimise sum of squares over; by default produces a sequence from 75 to 100 percent of sample size (in increments of 5 percent); a function is used here to facilitate varying sample sizes in cross-validation
int	a logical indicating whether to include an intercept
mio	one of 'min', 'all', or 'none' indicating whether to run the mixed-integer solver on the k and h that minimise the cv error, all k and h, or none at all
...	any other arguments (see rss.fit and rss.cv)

Details

The function is simply a wrapper that combines rss.fit and rss.cv to fit a sequence of robust subset selection models (indexed by k and h) and perform cross-validation for these models. Fitting is initially done using heuristics. Afterwards, the mixed-integer solver can be called to search for globally optimal solutions using the parameter mio. By default mio='min', indicating that the mixed integer solver will be run only on the tuning parameters that produced the minimum cross-validation error. It is possible to run the mixed-integer solver on all tuning parameters using mio='all', or not to call the solver at all using mio='none'.
See rss.fit and rss.cv for further options controlling the model fit and cross-validation.

Value

An object of class rss; a list with the following components:

cv	the output from rss.cv; see documentation
fit	the output from rss.fit; see documentation

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

Examples

```
# Generate training data with mixture error
set.seed(1)
n <- 100
p <- 10
p0 <- 5
n.c <- 5
beta <- c(rep(1, p0), rep(0, p - p0))
X <- matrix(rnorm(n * p), n, p)
e <- rnorm(n, c(rep(10, n.c), rep(0, n - n.c)))
y <- X %*% beta + e
```

```
# Fit robust subset selection models
# Run the mixed-integer solver on the (k,h) that minimises the cv error
fit <- rss(X, y, k = 0:10, h = function(n) round(c(0.95, 1.00) * n), mio = 'min', n.core = 1)

# Extract model coefficients and generate predictions
coef(fit)
predict(fit, X)

# Plot coefficient profiles and cross-validation results
plot(fit, type = 'profile')
plot(fit, type = 'cv')
```

`rss.cv` *Cross-validation for robust subset selection*

Description

Does (repeated) K-fold cross-validation for robust subset selection in parallel. In the interest of speed, uses heuristics without the mixed-integer solver.

Usage

```
rss.cv(
  X,
  y,
  k = 0:min(nrow(X) - int, ncol(X), 20),
  h = function(n) round(seq(0.75, 1, 0.05) * n),
  int = T,
  n.fold = 10,
  n.cv = 1,
  n.core = n.fold * n.cv,
  cv.objective = tmspe,
  ...
)
```

Arguments

X	a matrix of predictors
y	a vector of the response
k	the number of predictors to minimise sum of squares over; by default a sequence from 0 to 20
h	a function that takes the sample size that returns the number of observations to minimise sum of squares over; by default produces a sequence from 75 to 100 percent of sample size (in increments of 5 percent); a function is used here to facilitate varying sample sizes in cross-validation
int	a logical indicating whether to include an intercept

n.fold	the number of folds to use in cross-validation
n.cv	the number of times to repeat cross-validation; the results are averaged
n.core	the number of cores to use in cross-validation; by default n.fold * n.cv cores are used (if available)
cv.objective	the cross-validation objective function; by default trimmed mean square prediction error with 25 percent trimming
...	any other arguments

Value

An object of class rss.cv; a list with the following components:

mean.cv	a matrix with the cross-validated values of cv.objective; each row corresponds to a value of k and each column to a value of h
k.min	the k yielding the lowest cross-validated cv.objective
h.min	the h yielding the lowest cross-validated cv.objective
k	the value of k that was passed in
h	the value of h that was passed in

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

Examples

```
# Generate training data with mixture error
set.seed(1)
n <- 100
p <- 10
p0 <- 5
n.c <- 5
beta <- c(rep(1, p0), rep(0, p - p0))
X <- matrix(rnorm(n * p), n, p)
e <- rnorm(n, c(rep(10, n.c), rep(0, n - n.c)))
y <- X %*% beta + e

# Cross-validate robust subset selection models
cv <- rss.cv(X, y, k = 0:10, h = function(n) round(c(0.95, 1.00) * n), n.core = 1)

# Plot cross-validation results
plot(cv)
```

`rss.fit` *Fit a robust subset selection model*

Description

Fits a sequence of robust subset selection models using a combination of heuristics and mixed-integer optimisation (mio).

Usage

```
rss.fit(
  X,
  y,
  k = 0:min(nrow(X) - int, ncol(X), 20),
  h = round(seq(0.75, 1, 0.05) * nrow(X)),
  int = T,
  k.mio = NULL,
  h.mio = NULL,
  time = 60,
  tau = 1.25,
  output = F,
  params = NULL,
  robust = T,
  max.iter.ns = 100,
  max.iter.gd = 1e+05,
  tol = 1e-04,
  ...
)
```

Arguments

`X`	a matrix of predictors
`y`	a vector of the response
`k`	the number of predictors to minimise sum of squares over; by default a sequence from 0 to 20
`h`	the number of observations to minimise sum of squares over; by default a sequence from 75 to 100 percent of sample size (in increments of 5 percent)
`int`	a logical indicating whether to include an intercept
`k.mio`	the subset of `k` for which the mixed-integer solver should be run
`h.mio`	the subset of `h` for which the mixed-integer solver should be run
`time`	a time limit in seconds on each call to the mixed-integer solver
`tau`	a positive number greater than 1 used to tighten variable bounds in the mixed-integer formulation; small values give quicker run times but can also exclude the optimal solution

output	a logical indicating whether to print status updates
params	an optional list of additional Gurobi parameters (the parameters Time and OutputFlag are controlled by time and output)
robust	a logical indicating whether to standardise the data robustly; median/mad for true and mean/sd for false
max.iter.ns	the maximum number of neighbourhood search iterations to perform; if output is true then the number of iterations required for convergence will be printed
max.iter.gd	the maximum number of gradient descent iterations to perform
tol	a numerical tolerance parameter used to declare convergence
...	any other arguments

Details

The function first computes solutions over all combinations of k and h using heuristics. The heuristics include projected block-coordinate gradient descent and neighbourhood search (see our paper on arXiv). The solutions produced by the heuristics can be refined further using the mixed-integer solver. The tuning parameters that the solver operates on are specified by the k.mio and h.mio parameters, which must be subsets of k and h.
If robust is set to true and the median of any predictor is zero, then the data cannot be standardised (the median absolute deviation is undefined) and an error message will be returned.

Value

An object of class rss.fit; a list with the following components:

beta	an array of estimated regression coefficients; each column of regression coefficients corresponds to fixed value of k and each matrix to fixed value of h
weights	an array of binary weights; weights equal to one correspond to good observations selected for inclusion in the least squares fit; each column of weights corresponds to fixed value of k and each matrix to fixed value of h
objval	a matrix with the objective function values; each row corresponds to a value for different k and each column to a value for different h
k	the value of k that was passed in
h	the value of h that was passed in
int	whether an intercept was included

Author(s)

Ryan Thompson <ryan.thompson@monash.edu>

Examples

```
# Generate training data with mixture error
set.seed(1)
n <- 100
p <- 10
p0 <- 5
```

```
n.c <- 5
beta <- c(rep(1, p0), rep(0, p - p0))
X <- matrix(rnorm(n * p), n, p)
e <- rnorm(n, c(rep(10, n.c), rep(0, n - n.c)))
y <- X %*% beta + e

# Fit robust subset selection models and run the mixed-integer solver
fit <- rss.fit(X, y, k = 0:p, h = n - n.c, k.mio = 0:p, h.mio = n - n.c)

# Extract model coefficients and generate predictions
coef(fit, k = p0, h = n - n.c)
predict(fit, X, k = p0, h = n - n.c)

# Plot coefficient profiles
plot(fit)
```

Index