Package: splitSelect (via r-universe)

September 2, 2024

Type Package

Title Best Split Selection Modeling for Low-Dimensional Data

Version 1.0.3

Date 2021-11-08

Author Anthony Christidis <`anthony.christidis@stat.ubc.ca`>, Stefan Van Aelst <`stefan.vanaelst@kuleuven.be`>, Ruben Zamar <`ruben@stat.ubc.ca`>

Maintainer Anthony Christidis <`anthony.christidis@stat.ubc.ca`>

Description Functions to generate or sample from all possible splits of features or variables into a number of specified groups. Also computes the best split selection estimator (for low-dimensional data) as defined in Christidis, Van Aelst and Zamar (2019) <arXiv:1812.05678>.

License GPL (>= 2)

Biarch true

Imports multicool, glmnet, parallel, doParallel, foreach

RoxygenNote 7.1.1

Suggests testthat, mvnfast

Repository https://anthonychristidis.r-universe.dev

RemoteUrl https://github.com/anthonychristidis/splitselect

RemoteRef HEAD

RemoteSha d1b98105a54e6433342cc1c0d3af5de98c2da267

Contents

`coef.cv.splitSelect` *Coefficients for splitSelect object*

Description

`coef.cv.splitSelect` returns the coefficients for a cv.splitSelect for new data.

Usage

```
## S3 method for class 'cv.splitSelect'
coef(object, optimal.only = TRUE, ...)
```

Arguments

`object`	An object of class cv.splitSelect.
`optimal.only`	A boolean variable (TRUE default) to indicate if only the coefficient of the optimal split are returned.
`...`	Additional arguments for compatibility.

Value

A matrix with the coefficients of the `cv.splitSelect` object.

Author(s)

Anthony-Alexander Christidis, <`anthony.christidis@stat.ubc.ca`>

See Also

`splitSelect`

Examples

```
# Setting the parameters
p <- 4
n <- 30
n.test <- 5000
beta <- rep(5,4)
rho <- 0.1
r <- 0.9
```

```
SNR <- 3
# Creating the target matrix with "kernel" set to rho
target_cor <- function(r, p){
  Gamma <- diag(p)
  for(i in 1:(p-1)){
    for(j in (i+1):p){
      Gamma[i,j] <- Gamma[j,i] <- r^(abs(i-j))
    }
  }
  return(Gamma)
}
# AR Correlation Structure
Sigma.r <- target_cor(r, p)
Sigma.rho <- target_cor(rho, p)
sigma.epsilon <- as.numeric(sqrt((t(beta) %*% Sigma.rho %*% beta)/SNR))
# Simulate some data
x.train <- mvnfast::rmvn(30, mu=rep(0,p), sigma=Sigma.r)
y.train <- 1 + x.train %*% beta + rnorm(n=n, mean=0, sd=sigma.epsilon)

# Generating the coefficients for a fixed split

split.out <- cv.splitSelect(x.train, y.train, G=2, use.all=TRUE,
                            fix.partition=list(matrix(c(2,2),
                                                      ncol=2, byrow=TRUE)),
                            fix.split=NULL,
                            intercept=TRUE, group.model="glmnet", alphas=0, nfolds=10)
coef(split.out)
```

`coef.splitSelect` *Coefficients for splitSelect object*

Description

`coef.splitSelect` returns the coefficients for a splitSelect object.

Usage

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

Arguments

`object`	An object of class splitSelect.
`...`	Additional arguments for compatibility.

Value

A matrix with the coefficients of the `splitSelect` object.

Author(s)

Anthony-Alexander Christidis, <`anthony.christidis@stat.ubc.ca`>

See Also

`splitSelect`

Examples

```
# Setting the parameters
p <- 4
n <- 30
n.test <- 5000
beta <- rep(5,4)
rho <- 0.1
r <- 0.9
SNR <- 3
# Creating the target matrix with "kernel" set to rho
target_cor <- function(r, p){
  Gamma <- diag(p)
  for(i in 1:(p-1)){
    for(j in (i+1):p){
      Gamma[i,j] <- Gamma[j,i] <- r^(abs(i-j))
    }
  }
  return(Gamma)
}
# AR Correlation Structure
Sigma.r <- target_cor(r, p)
Sigma.rho <- target_cor(rho, p)
sigma.epsilon <- as.numeric(sqrt((t(beta) %*% Sigma.rho %*% beta)/SNR))
# Simulate some data
x.train <- mvnfast::rmvn(30, mu=rep(0,p), sigma=Sigma.r)
y.train <- 1 + x.train %*% beta + rnorm(n=n, mean=0, sd=sigma.epsilon)

# Generating the coefficients for a fixed partition of the variables

split.out <- splitSelect(x.train, y.train, G=2, use.all=TRUE,
                  fix.partition=list(matrix(c(2,2), ncol=2, byrow=TRUE)), fix.split=NULL,
                         intercept=TRUE, group.model="glmnet", alphas=0)
coef(split.out)
```

`cv.splitSelect` *Split Selection Modeling for Low-Dimensional Data - Cross-Validation*

Description

`cv.splitSelect` performs the best split selection algorithm with cross-validation

Usage

```
cv.splitSelect(
  x,
  y,
  intercept = TRUE,
  G,
  use.all = TRUE,
  family = c("gaussian", "binomial")[1],
  group.model = c("glmnet", "LS", "Logistic")[1],
  alphas = 0,
  nsample = NULL,
  fix.partition = NULL,
  fix.split = NULL,
  nfolds = 10,
  parallel = FALSE,
  cores = getOption("mc.cores", 2L)
)
```

Arguments

`x`	Design matrix.
`y`	Response vector.
`intercept`	Boolean variable to determine if there is intercept (default is TRUE) or not.
`G`	Number of groups into which the variables are split. Can have more than one value.
`use.all`	Boolean variable to determine if all variables must be used (default is TRUE).
`family`	Description of the error distribution and link function to be used for the model. Must be one of "gaussian" or "binomial".
`group.model`	Model used for the groups. Must be one of "glmnet" or "LS".
`alphas`	Elastic net mixing parameter. Should be between 0 (default) and 1.
`nsample`	Number of sample splits for each value of G. If NULL, then all splits will be considered (unless there is overflow).
`fix.partition`	Optional list with G elements indicating the partitions (in each row) to be considered for the splits.
`fix.split`	Optional matrix with p columns indicating the groups (in each row) to be considered for the splits.
`nfolds`	Number of folds for the cross-validation procedure.
`parallel`	Boolean variable to determine if parallelization of the function. Default is FALSE.
`cores`	Number of cores for the parallelization for the function.

Value

An object of class cv.splitSelect.

Author(s)

Anthony-Alexander Christidis, <`anthony.christidis@stat.ubc.ca`>

See Also

`coef.cv.splitSelect`, `predict.cv.splitSelect`

Examples

```
# Setting the parameters
p <- 4
n <- 30
n.test <- 5000
beta <- rep(5,4)
rho <- 0.1
r <- 0.9
SNR <- 3
# Creating the target matrix with "kernel" set to rho
target_cor <- function(r, p){
  Gamma <- diag(p)
  for(i in 1:(p-1)){
    for(j in (i+1):p){
      Gamma[i,j] <- Gamma[j,i] <- r^(abs(i-j))
    }
  }
  return(Gamma)
}
# AR Correlation Structure
Sigma.r <- target_cor(r, p)
Sigma.rho <- target_cor(rho, p)
sigma.epsilon <- as.numeric(sqrt((t(beta) %*% Sigma.rho %*% beta)/SNR))
# Simulate some data
x.train <- mvnfast::rmvn(30, mu=rep(0,p), sigma=Sigma.r)
y.train <- 1 + x.train %*% beta + rnorm(n=n, mean=0, sd=sigma.epsilon)

# Generating the coefficients for a fixed partition of the variables

split.out <- cv.splitSelect(x.train, y.train, G=2, use.all=TRUE,
                            fix.partition=list(matrix(c(2,2),
                                                      ncol=2, byrow=TRUE)),
                            fix.split=NULL,
                            intercept=TRUE, group.model="glmnet", alphas=0, nfolds=10)
```

`generate_partitions` *Generate Splits Partitions Possibilities*

Description

`generate_partitions` returns a matrix with the number of possible objects in each group using splits.

Usage

```
generate_partitions(p, G, use.all = TRUE)
```

Arguments

`p`	Number of variables or objects to split.
`G`	Number of groups into which the variables are split.
`use.all`	Boolean variable to determine if all variables must be used (default is TRUE).

Value

A matrix or list with the number of possible objects in each group using splits.

Author(s)

Anthony-Alexander Christidis, <`anthony.christidis@stat.ubc.ca`>

Examples

```
# Generating the possible split partitions of 6 variables in 3 groups
# Using all the variables
split.3groups.all <- generate_partitions(6, 3)
split.3groups.all
# Without using all the variables
split.3groups <- generate_partitions(6, 3, use.all=FALSE)
split.3groups
```

`generate_splits` *Generate Splits Possibilities*

Description

`generate_splits` returns a matrix with the different splits of the variables in reach row.

Usage

```
generate_splits(p, G, use.all = TRUE, fix.partition = NULL, verbose = TRUE)
```

Arguments

`p`	Number of variables or objects to split.
`G`	Number of groups into which the variables are split.
`use.all`	Boolean variable to determine if all variables must be used (default is TRUE).
`fix.partition`	Optional matrix with G columns (or list if more than one value of G) indicating the partitions (in each row) to be considered for the splits.
`verbose`	Boolean variable to determine if console output for cross-validation progress is printed (default is TRUE).

Value

A matrix with the different splits of the variables in the groups.

Author(s)

Anthony-Alexander Christidis, `<anthony.christidis@stat.ubc.ca>`

Examples

```
# Generating the possible splits of 6 variables in 3 groups
# Using all the variables
split.3groups.all <- generate_splits(6, 3)
split.3groups.all
# Without using all the variables
split.3groups <- generate_splits(6, 3, use.all=FALSE)
split.3groups
```

`nsplit` *Compute Total Number of Possible Splits*

Description

`nsplits` returns the total number of possible splits of variables into groups.

Usage

```
nsplit(p, G, use.all = TRUE, fix.partition = NULL)
```

Arguments

`p`	Number of variables or objects to split.
`G`	Number of groups into which the variables are split.
`use.all`	Boolean variable to determine if all variables must be used (default is TRUE).
`fix.partition`	Optional matrix with G columns (or list if more than one value of G) indicating the partitions (in each row) to be considered for the splits.

Value

A numeric vector with the total number of possible splits.

Author(s)

Anthony-Alexander Christidis, `<anthony.christidis@stat.ubc.ca>`

Examples

```
# Compute the total number of possible splits of 6 variables into 3 groups
# We use all the variables
out.n.splits.all <- nsplit(p=6, G=3, use.all=TRUE)
out.n.splits.all
# We don't enforce using all the variables
out.n.splits <- nsplit(p=6, G=3, use.all=FALSE)
out.n.splits
```

`predict.cv.splitSelect` *Predictions for cv.splitSelect object*

Description

`predict.cv.splitSelect` returns the prediction for cv.splitSelect for new data.

Usage

```
## S3 method for class 'cv.splitSelect'
predict(object, newx, optimal.only = TRUE, ...)
```

Arguments

`object`	An object of class cv.splitSelect.
`newx`	A matrix with the new data.
`optimal.only`	A boolean variable (TRUE default) to indicate if only the predictions of the optimal split are returned.
`...`	Additional arguments for compatibility.

Value

A matrix with the predictions of the `cv.splitSelect` object.

Author(s)

Anthony-Alexander Christidis, <`anthony.christidis@stat.ubc.ca`>

See Also

`cv.splitSelect`

Examples

```
# Setting the parameters
p <- 4
n <- 30
n.test <- 5000
beta <- rep(5,4)
rho <- 0.1
r <- 0.9
SNR <- 3
# Creating the target matrix with "kernel" set to rho
target_cor <- function(r, p){
  Gamma <- diag(p)
  for(i in 1:(p-1)){
    for(j in (i+1):p){
```

```
      Gamma[i,j] <- Gamma[j,i] <- r^(abs(i-j))
    }
  }
  return(Gamma)
}
# AR Correlation Structure
Sigma.r <- target_cor(r, p)
Sigma.rho <- target_cor(rho, p)
sigma.epsilon <- as.numeric(sqrt((t(beta) %*% Sigma.rho %*% beta)/SNR))
# Simulate some data
x.train <- mvnfast::rmvn(30, mu=rep(0,p), sigma=Sigma.r)
y.train <- 1 + x.train %*% beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma.rho)
y.test <- 1 + x.test %*% beta + rnorm(n.test, sd=sigma.epsilon)

# Generating the coefficients for a fixed split

split.out <- cv.splitSelect(x.train, y.train, G=2, use.all=TRUE,
                            fix.partition=list(matrix(c(2,2),
                                                      ncol=2, byrow=TRUE)),
                            fix.split=NULL,
                            intercept=TRUE, group.model="glmnet", alphas=0)
predict(split.out, newx=x.test)
```

`predict.splitSelect` *Predictions for splitSelect object*

Description

`predict.splitSelect` returns the prediction for splitSelect for new data.

Usage

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

Arguments

`object`	An object of class splitSelect.
`newx`	A matrix with the new data.
`...`	Additional arguments for compatibility.

Value

A matrix with the predictions of the `splitSelect` object.

Author(s)

Anthony-Alexander Christidis, <`anthony.christidis@stat.ubc.ca`>

See Also

`splitSelect`

Examples

```
# Setting the parameters
p <- 4
n <- 30
n.test <- 5000
beta <- rep(5,4)
rho <- 0.1
r <- 0.9
SNR <- 3
# Creating the target matrix with "kernel" set to rho
target_cor <- function(r, p){
  Gamma <- diag(p)
  for(i in 1:(p-1)){
    for(j in (i+1):p){
      Gamma[i,j] <- Gamma[j,i] <- r^(abs(i-j))
    }
  }
  return(Gamma)
}
# AR Correlation Structure
Sigma.r <- target_cor(r, p)
Sigma.rho <- target_cor(rho, p)
sigma.epsilon <- as.numeric(sqrt((t(beta) %*% Sigma.rho %*% beta)/SNR))
# Simulate some data
x.train <- mvnfast::rmvn(30, mu=rep(0,p), sigma=Sigma.r)
y.train <- 1 + x.train %*% beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma.rho)
y.test <- 1 + x.test %*% beta + rnorm(n.test, sd=sigma.epsilon)

# Generating the coefficients for a fixed split

split.out <- splitSelect(x.train, y.train, G=2, use.all=TRUE,
                  fix.partition=list(matrix(c(2,2), ncol=2, byrow=TRUE)), fix.split=NULL,
                         intercept=TRUE, group.model="glmnet", alphas=0)
predict(split.out, newx=x.test)
```

`rsplit` *Generate Samples of Splits Possibilities*

Description

`rsplit` returns a matrix with random splits of the variables in groups.

Usage

```
rsplit(n, p, G, use.all = TRUE, fix.partition = NULL, verbose = TRUE)
```

Arguments

`n`	Number of sample splits.
`p`	Number of variables or objects to split.
`G`	Number of groups into which the variables are split.
`use.all`	Boolean variable to determine if all variables must be used (default is TRUE).
`fix.partition`	Optional matrix with G columns indicating the partitions (in each row) to be considered for the splits.
`verbose`	Boolean variable to determine if console output for cross-validation progress is printed (default is TRUE).

Value

A matrix or list with the number of possible objects in each group using splits.

Author(s)

Anthony-Alexander Christidis, `<anthony.christidis@stat.ubc.ca>`

Examples

```
# Generating sample splits of 6 variables in 3 groups
# Using all the variables
random.splits <- rsplit(100, 6, 3)
# Using fixed partitions
random.splits.fixed <- rsplit(100, 6, 3, fix.partition=matrix(c(2,2,2), nrow=1))
```

`splitSelect` *Best Split Selection Modeling for Low-Dimensional Data*

Description

`splitSelect` performs the best split selection algorithm.

Usage

```
splitSelect(
  x,
  y,
  intercept = TRUE,
  G,
  use.all = TRUE,
  family = c("gaussian", "binomial")[1],
  group.model = c("glmnet", "LS", "Logistic")[1],
  lambdas = NULL,
  alphas = 0,
  nsample = NULL,
  fix.partition = NULL,
  fix.split = NULL,
  parallel = FALSE,
  cores = getOption("mc.cores", 2L),
  verbose = TRUE
)
```

Arguments

`x`	Design matrix.
`y`	Response vector.
`intercept`	Boolean variable to determine if there is intercept (default is TRUE) or not.
`G`	Number of groups into which the variables are split. Can have more than one value.
`use.all`	Boolean variable to determine if all variables must be used (default is TRUE).
`family`	Description of the error distribution and link function to be used for the model. Must be one of "gaussian" or "binomial".
`group.model`	Model used for the groups. Must be one of "glmnet" or "LS".
`lambdas`	The shinkrage parameters for the "glmnet" regularization. If NULL (default), optimal values are chosen.
`alphas`	Elastic net mixing parameter. Should be between 0 (default) and 1.
`nsample`	Number of sample splits for each value of G. If NULL, then all splits will be considered (unless there is overflow).
`fix.partition`	Optional list with G elements indicating the partitions (in each row) to be considered for the splits.
`fix.split`	Optional matrix with p columns indicating the groups (in each row) to be considered for the splits.
`parallel`	Boolean variable to determine if parallelization of the function. Default is FALSE.
`cores`	Number of cores for the parallelization for the function.
`verbose`	Boolean variable to determine if console output for cross-validation progress is printed (default is TRUE).

Value

An object of class splitSelect.

Author(s)

Anthony-Alexander Christidis, <`anthony.christidis@stat.ubc.ca`>

See Also

`coef.splitSelect`, `predict.splitSelect`

Examples

```
# Setting the parameters
p <- 4
n <- 30
n.test <- 5000
beta <- rep(5,4)
rho <- 0.1
r <- 0.9
SNR <- 3
# Creating the target matrix with "kernel" set to rho
target_cor <- function(r, p){
  Gamma <- diag(p)
  for(i in 1:(p-1)){
    for(j in (i+1):p){
      Gamma[i,j] <- Gamma[j,i] <- r^(abs(i-j))
    }
  }
  return(Gamma)
}
# AR Correlation Structure
Sigma.r <- target_cor(r, p)
Sigma.rho <- target_cor(rho, p)
sigma.epsilon <- as.numeric(sqrt((t(beta) %*% Sigma.rho %*% beta)/SNR))
# Simulate some data
x.train <- mvnfast::rmvn(30, mu=rep(0,p), sigma=Sigma.r)
y.train <- 1 + x.train %*% beta + rnorm(n=n, mean=0, sd=sigma.epsilon)

# Generating the coefficients for a fixed partition of the variables

split.out <- splitSelect(x.train, y.train, G=2, use.all=TRUE,
                         fix.partition=list(matrix(c(2,2),
                                                   ncol=2, byrow=TRUE)),
                         fix.split=NULL,
                         intercept=TRUE, group.model="glmnet", alphas=0)
```

`splitSelect_coef` *Split Selection for Regression - Coefficients Generation*

Description

`splitSelect_coef` generates the coefficients for a particular split of variables into groups.

Usage

```
splitSelect_coef(
  x,
  y,
  variables.split,
  intercept = TRUE,
  family = c("gaussian", "binomial")[1],
  group.model = c("glmnet", "LS", "Logistic")[1],
  lambdas = NULL,
  alphas = 0
)
```

Arguments

`x`	Design matrix.
`y`	Response vector.
`variables.split`	A vector with the split of the variables into groups as values.
`intercept`	Boolean variable to determine if there is intercept (default is TRUE) or not.
`family`	Description of the error distribution and link function to be used for the model. Must be one of "gaussian" or "binomial".
`group.model`	Model used for the groups. Must be one of "glmnet" or "LS".
`lambdas`	The shinkrage parameters for the "glmnet" regularization. If NULL (default), optimal values are chosen.
`alphas`	Elastic net mixing parameter. Should be between 0 (default) and 1.

Value

A vector with the regression coefficients for the split.

Author(s)

Anthony-Alexander Christidis, <`anthony.christidis@stat.ubc.ca`>

Examples

```
# Setting the parameters
p <- 6
n <- 30
n.test <- 5000
group.beta <- -3
beta <- c(rep(1, 2), rep(group.beta, p-2))
rho <- 0.1
r <- 0.9
SNR <- 3
# Creating the target matrix with "kernel" set to rho
target_cor <- function(r, p){
  Gamma <- diag(p)
  for(i in 1:(p-1)){
    for(j in (i+1):p){
      Gamma[i,j] <- Gamma[j,i] <- r^(abs(i-j))
    }
  }
  return(Gamma)
}
# AR Correlation Structure
Sigma.r <- target_cor(r, p)
Sigma.rho <- target_cor(rho, p)
sigma.epsilon <- as.numeric(sqrt((t(beta) %*% Sigma.rho %*% beta)/SNR))
# Simulate some data
x.train <- mvnfast::rmvn(30, mu=rep(0,p), sigma=Sigma.r)
y.train <- 1 + x.train %*% beta + rnorm(n=n, mean=0, sd=sigma.epsilon)
x.test <- mvnfast::rmvn(n.test, mu=rep(0,p), sigma=Sigma.rho)
y.test <- 1 + x.test %*% beta + rnorm(n.test, sd=sigma.epsilon)

# Generating the coefficients for a fixed split
splitSelect_coef(x.train, y.train, variables.split=matrix(c(1,2,1,2,1,2), nrow=1))
```

Index