Package 'SPSP'

February 22, 2022

Type Package

Title Selection by Partitioning the Solution Paths

Description

An implementation of the feature Selection procedure by Partitioning the entire Solution Paths (namely SPSP) to identify the relevant features rather than using a single tuning parameter. By utilizing the entire solution paths, this procedure can obtain better selection accuracy than the commonly used approach of selecting only one tuning parameter based on existing criteria, cross-validation (CV), generalized CV, AIC, BIC, and extended BIC (Liu, Y., & Wang, P. (2018) <doi:10.1214/18-EJS1434>). It is more stable and accurate (low false positive and false negative rates) than other variable selection approaches. In addition, it can be flexibly coupled with the solution paths of Lasso, adaptive Lasso, ridge regression, and other penalized estimators.

Version 0.1.1

Date 2022-02-21

Maintainer Xiaorui (Jeremy) Zhu <zhuxiaorui1989@gmail.com>

URL https://github.com/XiaoruiZhu/SPSP

BugReports https://github.com/XiaoruiZhu/SPSP/issues

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 3.5.0), glmnet, lars

Imports Rcpp (>= 1.0.7)

LinkingTo Rcpp

Suggests testthat (>= 3.0.0), MASS

Config/testthat/edition 3

RoxygenNote 7.1.2

NeedsCompilation yes

Author Xiaorui (Jeremy) Zhu [aut, cre],
Yang Liu [aut],
Peng Wang [aut]

Repository CRAN

Date/Publication 2022-02-22 00:58:48 UTC

R topics documented:

SPSP-package   *Selection by Partitioning the Solution Paths*

Description

An implementation of the feature Selection procedure by Partitioning the entire Solution Paths (namely SPSP) to identify the relevant features rather than using a single tuning parameter. By utilizing the entire solution paths, this procedure can obtain better selection accuracy than the commonly used approach of selecting only one tuning parameter based on existing criteria, cross-validation (CV), generalized CV, AIC, BIC, and EBIC (Liu, Y., & Wang, P. (2018)). It is more stable and accurate (low false positive and false negative rates) than other variable selection approaches. In addition, it can be flexibly coupled with the solution paths of Lasso, adaptive Lasso, ridge regression, and other penalized estimators.

Details

This package includes two main functions and several functions (fitfun.SP) to obtains the solution paths. The SPSP function allows users to specify the penalized likelihood approaches that will generate the solution paths for the SPSP procedure. Then this function will automatically partitioning the entire solution paths. Its key idea is to classify variables as relevant or irrelevant at each tuning parameter and then to select all of the variables which have been classified as relevant at least once. The SPSP_step purely apply the partitioning step that needs the solution paths as the input. In addition, there are several functions to obtain the solution paths. They can be used as an input of fitfun.SP argument.

Author(s)

Xiaorui (Jeremy) Zhu, <zhuxiaorui1989@gmail.com>,
Yang Liu, <yliu23@fhcrc.org>,
Peng Wang, <wangp9@ucmail.uc.edu>

References

Liu, Y., & Wang, P. (2018). Selection by partitioning the solution paths. *Electronic Journal of Statistics*, 12(1), 1988-2017. <10.1214/18-EJS1434>

Fitting-Functions	*Four Fitting-Functions that can be used as an input of* fitfun.SP *argument to obtain the solution paths for the SPSP algorithm. The users can also customize a function to generate the solution paths. As long as the customized function take arguments x, y, family, standardize, and intercept, and return an object of class* glmnet, lars *(or* SCAD, MCP *in the future).*

Description

Four Fitting-Functions that can be used as an input of fitfun.SP argument to obtain the solution paths for the SPSP algorithm. The users can also customize a function to generate the solution paths. As long as the customized function take arguments x, y, family, standardize, and intercept, and return an object of class glmnet, lars (or SCAD, MCP in the future).

lasso.glmnet uses lasso selection from glmnet.

adalasso.glmnet the function to conduct the adaptive lasso selection using the lambda.1se from cross-validation lasso method to obtain initial coefficients. It uses package glmnet.

adalassoCV.glmnet adaptive lasso selection using the lambda.1se from cross-validation adaptive lasso method to obtain initial coefficients. It uses package glmnet.

ridge.glmnet uses ridge regression to obtain the solution path.

lasso.lars uses lasso selection in lars to obtain the solution path.

Usage

```
lasso.glmnet(x, y, family, standardize, intercept, ...)

adalasso.glmnet(x, y, family, standardize, intercept, ...)

adalassoCV.glmnet(x, y, family, standardize, intercept, ...)

ridge.glmnet(x, y, family, standardize, intercept, ...)

lasso.lars(x, y, family, standardize, intercept, ...)
```

Arguments

x	a matrix of the independent variables. The dimensions are (nobs) and (nvars); each row is an observation vector.
y	Response variable. Quantitative for family="gaussian" or family="poisson" (non-negative counts). For family="binomial" should be either a factor with two levels.
family	Response type. Either a character string representing one of the built-in families, or else a glm() family object.
standardize	logical argument. Should conduct standardization before the estimation? Default is TRUE.

| intercept | logical. If x is a data.frame, this argument determines if the resulting model matrix should contain a separate intercept or not. Default is TRUE. |
| ... | Additional optional arguments. |

Value

An object of class `"glmnet"` is returned to provide solution paths for the SPSP algorithm.

An object of class `"glmnet"` is returned to provide solution paths for the SPSP algorithm.

An object of class `"glmnet"` is returned to provide solution paths for the SPSP algorithm.

An object of class `"glmnet"` is returned to provide solution paths for the SPSP algorithm.

An object of class `"lars"` is returned to provide solution paths for the SPSP algorithm.

| HighDim | *A high dimensional dataset with n equals to 200 and p equals to 500.* |

Description

A dataset with 200 observations and 500 dimensions is generated from the following process: linear regression model with only first three non-zero coefficients equal to 3, 2, and 1.5 respectively. The covariates are correlated with AR structure (rho=0.3). The error term is normally distributed with zero mean and sd equals to 0.5.

Usage

```
data(HighDim)
```

Examples

```
# HighDim dataset is generated from the following process:
n <- 200; p <- 500; sigma <- 0.5
beta <- rep(0, p); nonzero <- c(1, 2, 3); zero <- setdiff(1:p, nonzero)
beta[nonzero] <- c(3, 2, 1.5)
Sigma <- 0.3^(abs(outer(1:p,1:p,"-")))
library(MASS)
X <- mvrnorm(n, rep(0,p), Sigma)
error <- rnorm(n, 0, sigma)

X <- apply(X, 2, scale) * sqrt(n)/sqrt(n-1)
error <- error - mean(error)

Y <- X %*% beta + error
HighDim <- data.frame(Y, X)
head(HighDim)
```

Description

A user-friendly function to conduct the selection by Partitioning the Solution Paths (the SPSP algorithm). The user only needs to specify the independent variables matrix, response, family, and `fitfun.SP`.

Usage

```
SPSP(
  x,
  y,
  family = c("gaussian", "binomial"),
  fitfun.SP = lasso.glmnet,
  args.fitfun.SP = list(),
  standardize = TRUE,
  intercept = TRUE,
  ...
)
```

Arguments

x	A matrix with all independent variables, of dimension n by p; each row is an observation vector with p variables.
y	Response variable. Quantitative for `family="gaussian"` or `family="poisson"` (non-negative counts). For `family="binomial"` should be either a factor with two levels.
family	Response type. Either a character string representing one of the built-in families, or else a glm() family object.
fitfun.SP	A function to obtain the solution paths for the SPSP algorithm. This function takes the arguments x, y, family as above, and additionally the standardize and intercept and others in [glmnet](#) or [lars](#). The function fit the model with lasso, adaptive lasso, or ridge regression to return the solution path of the corresponding penalized likelihood approach.
	`lasso.glmnet` lasso selection from [glmnet](#).
	`adalasso.glmnet` adaptive lasso selection using the `lambda.1se` from cross-validation lasso method to obtain initial coefficients. It uses package [glmnet](#).
	`adalassoCV.glmnet` adaptive lasso selection using the `lambda.1se` from cross-validation adaptive lasso method to obtain initial coefficients. It uses package [glmnet](#).
	`ridge.glmnet` use ridge regression to obtain the solution path.
	`lasso.lars` use lasso selection in [lars](#) to obtain the solution path.

args.fitfun.SP	A named list containing additional arguments that are passed to the fitting function; see also argument `args.fitfun.SP` in do.call.
standardize	logical argument. Should conduct standardization before the estimation? Default is TRUE.
intercept	logical. If x is a data.frame, this argument determines if the resulting model matrix should contain a separate intercept or not. Default is TRUE.
...	Additional optional arguments.

Value

An object of class `"SPSP"` is a list containing at least the following components:

beta_SPSP	the estimated coefficients of SPSP selected model;
S0	the estimated relevant sets;
nonzero	the selected covariates;
zero	the covariates that are not selected;
thres	the boundaries for abs(beta);
R	the sorted adjacent distances;
intercept	the estimated intercept when `intercept == T`.

This object has attribute contains:

glmnet.fit	the fitted penalized regression within the input function `fitfun.SP`;
family	the family of fitted object;
fitfun.SP	the function to obtain the solution paths for the SPSP algorithm;
args.fitfun.SP	a named list containing additional arguments for the function `fitfun.SP`.

Examples

```
data(HighDim)
library(glmnet)
# Use the high dimensional dataset (data(HighDim)) to test SPSP+Lasso and SPSP+AdaLasso:
data(HighDim)

x <- as.matrix(HighDim[,-1])
y <- HighDim[,1]

spsp_lasso_1 <- SPSP::SPSP(x = x, y = y, family = "gaussian", fitfun.SP = lasso.glmnet,
                           init = 1, standardize = FALSE, intercept = FALSE)

head(spsp_lasso_1$nonzero)
head(spsp_lasso_1$beta_SPSP)

spsp_adalasso_5 <- SPSP::SPSP(x = x, y = y, family = "gaussian", fitfun.SP = adalasso.glmnet,
                              init = 5, standardize = TRUE, intercept = FALSE)

head(spsp_adalasso_5$nonzero)
head(spsp_adalasso_5$beta_SPSP)
```

SPSP_step   *The selection step with the input of the solution paths.*

Description

A function to select the relevant predictors by partitioning the solution paths (the SPSP algorithm) based on the user provided solution paths BETA.

Usage

```
SPSP_step(
  x,
  y,
  family = c("gaussian", "binomial"),
  BETA,
  standardize = TRUE,
  intercept = TRUE,
  init = 1,
  R = NULL,
  ...
)
```

Arguments

x	independent variables as a matrix, of dimension nobs x nvars; each row is an observation vector.
y	response variable. Quantitative for family="gaussian" or family="poisson" (non-negative counts). For family="binomial" should be either a factor with two levels.
family	either a character string representing one of the built-in families, or else a glm() family object.
BETA	the solution paths obtained from a prespecified fitting step fitfun.SP = lasso.glmnet etc. It must be a p by k matrix, should be thicker and thicker, each column corresponds to a lambda, and lambda gets smaller and smaller. It is just the returned value beta from a glmnet object.
standardize	whether need standardization.
intercept	logical. If x is a data.frame, this argument determines if the resulting model matrix should contain a separate intercept or not.
init	initial coefficients, starting from init-th estimator of the solution paths. The default is 1.
R	sorted adjacent distances, default is NULL. Will be calculated inside.
...	Additional optional arguments.

Value

A list containing at least the following components:

beta_SPSP	the estimated coefficients of SPSP selected model;
S0	the estimated relevant sets;
nonzero	the selected covariates;
zero	the covariates that are not selected;
thres	the boundaries for abs(beta);
R	the sorted adjacent distances;
intercept	the estimated intercept when intercept == T.

This object has attribute contains:

glmnet.fit	the fitted penalized regression within the input function fitfun.SP;
family	the family of fitted object;
fitfun.SP	the function to obtain the solution paths for the SPSP algorithm;
args.fitfun.SP	a named list containing additional arguments for the function fitfun.SP.

Examples

```
data(HighDim)
library(glmnet)

x <- as.matrix(HighDim[,-1])
y <- HighDim[,1]

lasso_fit <- glmnet(x = x, y = y, alpha = 1, intercept = FALSE)

# SPSP+Lasso method
K <- dim(lasso_fit$beta)[2]
LBETA <- as.matrix(lasso_fit$beta)

spsp_lasso_1 <- SPSP_step(x = x, y = y, BETA = LBETA,
                          init = 1, standardize = FALSE, intercept = FALSE)
head(spsp_lasso_1$nonzero)
head(spsp_lasso_1$beta_SPSP)
```

Index