Package 'VariableScreening'

October 12, 2022

Type Package

Title High-Dimensional Screening for Semiparametric Longitudinal Regression

Version 0.2.1

Depends R (>= 3.2.1)

Description Implements variable screening techniques for ultra-high dimensional regression settings. Techniques for independent (iid) data, varying-coefficient models, and longitudinal data are implemented. The package currently contains three screen functions: screenIID(), screenLD() and screenVCM(), and six methods for simulating dataset: simulateDCSIS(), simulateLD, simulateMVSIS(), simulateMVSISNY(), simulateSIRS() and simulateVCM(). The package is based on the work of Li-Ping ZHU, Lexin LI, Runze LI, and Li-Xing ZHU (2011) <DOI:10.1198/jasa.2011.tm10563>, Runze LI, Wei ZHONG, & Liping ZHU (2012) <DOI:10.1080/01621459.2012.695654>, Jingyuan LIU, Runze LI, & Rongling WU (2014) <DOI:10.1080/01621459.2013.850086> Hengjian CUI, Runze LI, & Wei ZHONG (2015) <DOI:10.1080/01621459.2014.920256>, and Wanghuan CHU, Runze LI and Matthew REIMHERR (2016) <DOI:10.1214/16-AOAS912>.

Encoding UTF-8

Imports gee, expm, splines, MASS, energy

License GPL (>= 2)

RoxygenNote 7.2.0

NeedsCompilation no

Author Runze Li [aut],
Liying Huang [aut],
John Dziak [aut, cre]

Maintainer John Dziak <dziakj1@gmail.com>

Repository CRAN

Date/Publication 2022-06-23 22:20:02 UTC

R topics documented:

`screenIID` *Feature Selection for Ultrahigh-Dimensional Datasets with Independent Subjects,*

Description

Implements one of three screening procedures: Sure Independent Ranking and Screening (SIRS), Distance Correlation Sure Independence Screening (DC-SIS), or MV Sure Independence Screening (MV-SIS). In general they are extensions of the sure independence screening concept proposed by Fan and Lv (2008), but without a parametric assumption (e.g., linear or logistic) on the relationship between the predictor variables X and outcome Y.

Screening methods each rank the predictors based on some measure of their estimated strength of relationship with Y. The assumption is that only a few among the top-ranked variables are likely to be truly significant predictors.

The original version of SIS involved ranking the predictors by their correlation with Y, implying a linear relationship. The SIRS method is an extension proposed by Zhu, Li, Li, & Zhu (2011), which involved ranking the predictors by their correlation with the rank-ordered Y instead, thereby not assuming a linear correlation, and potentially outperforming SIS.

DC-SIS was then proposed by Li, Zhong and Zhu (2012) and its relationship measure is the distance correlation (DC) between a covariate and the outcome, a nonparametric generalization of the correlation coefficient (Szekely, Rizzo, & Bakirov, 2007). The function uses the dcor function from the R package energy in order to calculate this correlation. Simulations showed that DC-SIS could sometimes provide a further advantage over SIRS.

The above measures were primarily intended for a numerical Y. Cui, Li, and Zhong (2015) proposed MV-SIS, which was developed for categorical Y (including binary Y) as in discriminant analysis, and which is also robust to heavy-tailed predictor distributions. The measure used by MV-SIS for the association strength between a particular Xk and Y is a mean conditional variance measure called MV for short, namely the expectation in X of the variance in Y of the conditional cumulative distribution function F(x|Y)=P(X<=x|Y); note that like the correlation or distance correlation, this is zero if X and Y are independent because F(x) does not depend on Y in that case. Cui, Li, and Zhong (2015) also point out that the MV-SIS can alternatively be used with categorical X variables and numerical Y, instead of numerical X and categorical Y. This function supports that option as "MV-SIS-NY."

Whichever option is chosen, the function returns the ranking of the predictors according to the appropriate association measure.

The function code is adapted from the relevant authors' code. Special thanks are due to Wei Zhong for providing some of the code upon which this function is based.

Usage

```
screenIID(X, Y, method = "DC-SIS")
```

Arguments

X	Matrix of predictors to be screened. There should be one row for each observation.
Y	Vector of responses. It should have the same length as the number of rows of X. The responses should be numerical if SIRS or DC-SIS is used. The responses should be integers representing response categories if MV-SIS is used. Binary responses can be used for any method.
method	Screening method. The options are "SIRS", "DC-SIS", "MV-SIS" and "MV-SIS-NY", as described above.

Value

A list with following components:

measurement A vector of length equal to the number of columns in the input matrix X. It contains estimated strength of relationship with Y rank The rank of the error measures. This will have length equal to the number of columns in the input matrix X, and will consist of a permutation of the integers 1 through that length. A rank of 1 indicates the feature which appears to have the best predictive performance, 2 represents the second best and so on.

References

Cui, H., Li, R., & Zhong, W. (2015). Model-free feature screening for ultrahigh dimensional discriminant analysis. Journal of the American Statistical Association, 110: 630-641. <DOI:10.1080/01621459.2014.920256>

Fan, J., & Lv, J. (2008). Sure independence screening for ultrahigh dimensional feature space. Journal of the Royal Statistical Society, B, 70: 849-911. <DOI:10.1111/j.1467-9868.2008.00674.x>

Li, R., zhong, W., & Zhu, L. (2012). Feature screening via distance correlation learning. Journal of the American Statistical Association, 107: 1129-1139. <DOI:10.1080/01621459.2012.695654>

Szekely, G. J., Rizzo, M. L., & Bakirov, N. K. (2007). Measuring and Testing Dependence by Correlation of Distances. Annals of Statistics, 35, 2769-2794. <DOI: 10.1214/009053607000000505>

Zhu, L.-P., Li, L., Li, R., & Zhu, L.-X. (2011) Model-free feature screening for ultrahigh-dimensional data. Journal of the American Statistical Association, 106: 1464-1475. <DOI:10.1198/jasa.2011.tm10563>

Examples

```
set.seed(12345678)
results <- simulateDCSIS(n=100,p=500)
rank<- screenIID(X = results$X, Y = results$Y, method="DC-SIS")
```

screenLD *Perform high-dimensional screening for semiparametric longitudinal regression*

Description

Implements a screening procedure proposed by Chu, Li, and Reimherr (2016) <DOI:10.1214/16-AOAS912> for varying coefficient longitudinal models with ultra-high dimensional predictors. The effect of each predictor is allowed to vary over time, approximated by a low-dimensional B-spline. Within-subject correlation is handled using a generalized estimation equation approach with structure specified by the user. Variance is allowed to change over time, also approximated by a B-spline.

Usage

```
screenLD(
  X,
  Y,
  z,
  id,
  subset = 1:ncol(X),
  time,
  degree = 3,
  df = 4,
  corstr = "stat_M_dep",
  M = NULL
)
```

Arguments

X	Matrix of features (for example, SNP's). There should be one row for each observation.
Y	Vector of responses. It should have the same length as the number of rows of X.
z	Optional matrix of covariates to be included in all models. They may include demographic covariates such as gender or ethnic background, or some other theoretically important constructs. It should have the same number of rows as the number of rows of X. We suggest a fairly low dimensional z. If the model is intended to include an intercept function (which is recommended), then z should include a column of 1's representing the constant term.
id	Vector of integers identifying the subject to which each observation belongs. It should have the same length as the number of rows of X.
subset	Vector of integers identifying a subset of the features of X to be screened, the default is 1:ncol(X), i.e., to screen all columns of X.
time	Vector of real numbers identifying observation times. It should have the same length as the number of rows of X. We suggest using the convention of scaling time to the interval [0,1].

degree	Degree of the piecewise polynomial for the B-spline basis for the varying coefficient functions; see the documentation for the bs() function in the splines library.
df	Degrees of freedom of the B-spline basis for the varying coefficient functions; see the documentation for the bs() function in the splines library.
corstr	Working correlation structure for the generalized estimation equations model used to estimate the coefficient functions; see the documentation for the gee() function in the gee library. Options provided by the gee() function include "independence", "fixed", "stat_M_dep", "non_stat_M_dep", "exchangeable", "AR-M" and "unstructured".
M	An integer indexing the M value (complexity) of the dependence structure, if corstr is M-dependent or AR-M; see the documentation for the gee() function in the gee library. This will be ignored if the correlation structure does not require an M parameter. The default value is set to be 1.

Value

A list with following components: error A vector of length equal to the number of columns in the input matrix X. It contains sum squared error values for regression models which include the time-varying effects of the z covariates (if any) as well as each X covariate by itself. The lower this error is, the more desirable it is to retain the corresponding X covariate in a later predictive model. rank The rank of the error measures. This will have length equal to the number of columns in the input matrix X, and will consist of a permutation of the integers 1 through that length. A rank of 1 indicates the feature which appears to have the best predictive performance, 2 represents the second best and so on.

Examples

```
set.seed(12345678)
results <- simulateLD(p=500)
subset1 <- seq(1,5,2)
subset2 <- seq(100,200,2)
subset3 <- seq(202,400,2)
subset4 <- seq(401,499,2)
set <-c(subset1,subset2,subset3,subset4)
Jmin <- min(table(results$id)) - 1
screenResults <- screenLD(X = results$X,
                          Y = results$Y,
                          z = results$z,
                          id = results$id,
                          subset = set,
                          time = results$time,
                          degree = 3,
                          df = 4,
                          corstr = "stat_M_dep",
                          M = Jmin
                          )


rank <- screenResults$rank
```

```
unlist(rank)
trueIdx <- c(5,100,200,400)
rank[which(set %in% trueIdx)]
```

---

screenVCM *Perform screening for ultrahigh-dimensional varying coefficient model*

Description

Implements a screening procedure proposed by Liu, Li and Wu(2014) for varying coefficient models with ultra-high dimensional predictors.

The function code is adapted from the relevant authors' code. Special thanks are due to Jingyuan Liu for providing some of the code upon which this function is based.

Usage

```
screenVCM(X, Y, U)
```

Arguments

X	Matrix of predictors to be screened. There should be one row for each observation.
Y	Vector of responses. It should have the same length as the number of rows of X.
U	Covariate, with which coefficient functions vary.

Value

A list with following components: CORR_sq A vector of the unconditioned squared correlation with length equal to the number of columns in the input matrix X. The hgh the unconditioned squared correlation is, the more desirable it is to retain the corresponding X covariate in a later predictive model. rank Vector for the rank of the predictors in terms of the conditional correlation ($\hat{rho}*_j$ in the paper). This will have length equal to the number of columns in the input matrix X, and will consist of a permutation of the integers 1 through that length. A rank of 1 indicates the feature which appears to have the best marginal predictive performance with largest $\hat{rho}*_j$, 2 represents the second best and so forth.

References

Liu, J., Li, R., & Wu, R. (2014). Feature selection for varying coefficient models with ultrahigh-dimensional covariates. Journal of the American Statistical Association, 109: 266-274. <DOI:10.1080/01621459.2013.85008

Examples

```
set.seed(12345678)
results <- simulateVCM(p=400,
                       trueIdx = c(2, 100, 300),
                       betaFun = function(U) {
                         beta2 <- 2*I(U>0.4)
                         beta100 <- 1+U
                         beta300 <- (2-3*U)^2
                         return(c(beta2,
                                  beta100,
                                  beta300))
                         })
screenResults<- screenVCM(X = results$X,
                          Y = results$Y,
                          U = results$U)
rank <- screenResults$rank
unlist(rank)
trueIdx <- c(2,100,400, 600, 1000)
rank[trueIdx]
```

`simulateDCSIS` *Simulate a dataset for demonstrating the performance of screenIID with the DC-SIS method*

Description

Simulates a dataset that can be used to demonstrate variable screening for ultrahigh-dimensional regression with the DC-SIS option in screenIID. The simulated dataset has p numerical predictors X and a categorical Y-response. The data-generating scenario is a simplified version of Example 3.1a (homoskedastic) or 3.1d (heteroskedastic) of Li, Zhong & Zhu (2012). Specifically, the X covariates are normally distributed with mean zero and variance one, and may be correlated if the argument rho is set to a nonzero value. The response Y is generated as either Y = 6*X1 + 1.5*X2 + 9*1X12 < 0 + exp(2*X22)*e if heteroskedastic=TRUE, or Y = 6*X1 + 1.5*X2 + 9*1X12 < 0 + 6*X22 + e if heteroskedastic=FALSE, where e is a standard normal error term and 1 is a zero-one indicator function for the truth of the statement contained. Special thanks are due to Wei Zhong for providing some of the code upon which this function is based.

Usage

```
simulateDCSIS(n = 200, p = 5000, rho = 0, heteroskedastic = TRUE)
```

Arguments

`n`	Number of subjects in the dataset to be simulated. It will also equal to the number of rows in the dataset to be simulated, because it is assumed that each row represents a different independent and identically distributed subject.
`p`	Number of predictor variables (covariates) in the simulated dataset. These covariates will be the features screened by DC-SIS.

rho — The correlation between adjacent covariates in the simulated matrix X. The within-subject covariance matrix of X is assumed to has the same form as an AR(1) autoregressive covariance matrix, although this is not meant to imply that the X covariates for each subject are in fact a time series. Instead, it is just used as an example of a parsimonious but nontrivial covariance structure. If rho is left at the default of zero, the X covariates will be independent and the simulation will run faster.

heteroskedastic — Whether the error variance should be allowed to depend on one of the predictor variables.

Value

A list with following components: X Matrix of predictors to be screened. It will have n rows and p columns. Y Vector of responses. It will have length n.

References

Li, R., Zhong, W., & Zhu, L. (2012) Feature screening via distance correlation learning. Journal of the American Statistical Association, 107: 1129-1139. <DOI:10.1080/01621459.2012.695654>

Examples

```
set.seed(12345678)
results <- simulateDCSIS()
```

---

simulateLD *Simulate a dataset for testing the performance of screenlong*

Description

Simulates a dataset that can be used to test the screenlong function, and to test the performance of the proposed method under different scenarios. The simulated dataset has two z-covariates and p x-covariates, only a few of which have nonzero effect. There are n subjects in the simulated dataset, each having J observations, which are not necessarily evenly timed, we randomly draw a subset to create an unbalanced dataset. The within-subject correlation is assumed to be AR-1.

Usage

```
simulateLD(
  n = 100,
  J = 10,
  rho = 0.6,
  p = 500,
  trueIdx = c(5, 100, 200, 400),
  beta0Fun = NULL,
  betaFun = NULL,
  gammaFun = NULL,
```

```
  varFun = NULL
)
```

Arguments

n	Number of subjects in the simulated dataset
J	Number of observations per subject
rho	The correlation parameter for the AR-1 correlation structure.
p	The total number of features to be screened from
trueIdx	The indexes for the active features in the simulated x matrix. This should be a vector, and the values should be a subset of 1:p.
beta0Fun	The time-varying intercept for the data-generating model, as a function of time. If left as null, it will default to `f(t) 2 * t^2 - 1`. Time is assumed to be scaled to the interval [0,1].
betaFun	The time-varying coefficients for z in the data-generating model, as a function of time. If left as null, it will be specified as two functions. The first is `f(t) exp(t + 1)/2`. The second is `f(t) t^2 + 0.5`. Time is assumed to be scaled to the interval [0,1].
gammaFun	A list of functions of time, one function for each entry in trueIdx, giving the time-varying effects of each active feature in the simulated x matrix. If left as null, it will be specified as four functions. The first is a step function `f(t)=(t > 0.4)`. The second is `f(t)=- cos(2 * pi * t)`. the third is `f(t)=(2 - 3 * t)^2/2 - 1`. The fourth is `f(t)=sin(2 * pi * t)`.
varFun	A function of time telling the marginal variance of the error function at a given time. If left as null, it will be specified as `function(t) 0.5 + 3 * t^3`.

Value

A list with following components: x Matrix of features to be screened. It will have n*J rows and p columns. y Vector of responses. It will have length of n*J. z A matrix representing covariates to be included in each of the screening models. The first column will be all ones, representing the intercept. The second will consist of random ones and zeros, representing simulated genders. id Vector of integers identifying the subject to which each observation belongs. time Vector of real numbers identifying observation times. It should have the same length as the number of rows of x.

Examples

```
set.seed(12345678)
results <- simulateLD(p=1000)
```

`simulateMVSIS` *Simulate a dataset for demonstrating the performance of screenIID with the MV-SIS option with categorical outcome variable*

Description

Simulates a dataset that can be used to test screenIID for ultrahigh-dimensional discriminant analysis with the MV-SIS option. The simulation is based on the balanced scenarios in Example 3.1 of Cui, Li & Zhong (2015). The simulated dataset has p numerical X-predictors and a categorical Y-response. Special thanks are due to Wei Zhong for providing some of the code upon which this function is based.

Usage

```
simulateMVSIS(R = 2, n = 40, p = 2000, mu = 3, heavyTailedCovariates = FALSE)
```

Arguments

`R`	a positive integer, number of outcome categories for multinomial (categorical) outcome Y.
`n`	Number of subjects in the dataset to be simulated. It will also equal to the number of rows in the dataset to be simulated, because it is assumed that each row represents a different independent and identically distributed subject.
`p`	Number of predictor variables (covariates) in the simulated dataset. These covariates will be the features screened by DC-SIS.
`mu`	Signal strength; the larger mu is, the easier the active covariates will be to discover. # Specifically, mu is added to the rth predictor for r=1,...,R, so that the probability that Y equals r will be higher if the rth predictor is higher. It is assumed that p»r so that most predictors will be inactive. In real data there is no reason why, say, the first two columns in the matrix should be the important ones, but this is convenient in a simulation and the choice of permutation of the columns involves no loss of generality.
`heavyTailedCovariates`	If TRUE, the covariates will be generated as independent t variates, plus covariate-specific constants. If FALSE, they will be generated as independent standard normal variates.

Value

A list with following components: X Matrix of predictors to be screened. It will have n rows and p columns. Y Vector of responses. It will have length n.

References

Cui, H., Li, R., & Zhong, W. (2015). Model-free feature screening for ultrahigh dimensional discriminant analysis. Journal of the American Statistical Association, 110: 630-641. <DOI:10.1080/01621459.2014.920256>

Examples

```
set.seed(12345678)
results <- simulateMVSIS()
```

`simulateMVSISNY` *Simulate a dataset for demonstrating the performance of screenIID with the MV-SIS method with numeric outcome Y*

Description

Simulates a dataset that can be used to demonstrate variable screening for ultrahigh-dimensional regression with categorical predictors and numerical outcome variable using the MV-SIS-NY option in screenIID. The simulated dataset has p numerical predictors X and a categorical response Y. The X covariates are generated as binary with success probability 0.5 each. The response Y is generated as Y = 5*X1 + 5*X2 + 5*X12 + 5*X22 + e if heteroskedastic=FALSE, where e is a standard normal error term and 1 is a zero-one indicator function for the truth of the statement contained. Special thanks are due to Wei Zhong for providing some of the code upon which this function is based.

Usage

```
simulateMVSISNY(n = 500, p = 1000)
```

Arguments

`n`	Number of subjects in the dataset to be simulated. It will also equal to the number of rows in the dataset to be simulated, because it is assumed that each row represents a different independent and identically distributed subject.
`p`	Number of predictor variables (covariates) in the simulated dataset. These covariates will be the features screened by DC-SIS.

Value

A list with following components: X Matrix of predictors to be screened. It will have n rows and p columns. Y Vector of responses. It will have length n.

References

Cui, H., Li, R., & Zhong, W. (2015). Model-free feature screening for ultrahigh dimensional discriminant analysis. Journal of the American Statistical Association, 110: 630-641. <DOI:10.1080/01621459.2014.920256>

Examples

```
set.seed(12345678)
results <- simulateMVSISNY()
```

`simulateSIRS` *Simulate a dataset for demonstrating the performance of screenIID with the SIRS method*

Description

Simulates a dataset that can be used to demonstrate variable screening for ultrahigh-dimensional regression with the SIRS option in screenIID. The simulated dataset has p numerical predictors X and a categorical Y-response. The data-generating scenario is a simplified version of Example 1 of Zhu, Li, Li and Zhu (2011). Specifically, the X covariates are normally distributed with mean zero and variance one, and may be correlated if the argument rho is set to a nonzero value. The response Y is generated as Y = c*X1 + 0.8*c*X2 + 0.6*c*X3 + 0.4*c*X4 + 0.5*c*X5 + sigma*e. where c is the argument SignalStrength, e is either a standard normal distribution (if HeavyTailedResponse==FALSE) or t distribution with 1 degree of freedom (if HeavyTailedResponse==TRUE). sigma is either sqrt(6.83) if heteroskedastic==FALSE, or else exp(X20+X21+X22) if heteroskedastic=TRUE.

Usage

```
simulateSIRS(
  n = 200,
  p = 5000,
  rho = 0,
  HeavyTailedResponse = TRUE,
  heteroskedastic = TRUE,
  SignalStrength = 1
)
```

Arguments

`n`	Number of subjects in the dataset to be simulated. It will also equal to the number of rows in the dataset to be simulated, because it is assumed that each row represents a different independent and identically distributed subject.
`p`	Number of predictor variables (covariates) in the simulated dataset. These covariates will be the features screened by DC-SIS.
`rho`	The correlation between adjacent covariates in the simulated matrix X. The within-subject covariance matrix of X is assumed to has the same form as an AR(1) autoregressive covariance matrix, although this is not meant to imply that the X covariates for each subject are in fact a time series. Instead, it is just used as an example of a parsimonious but nontrivial covariance structure. If rho is left at the default of zero, the X covariates will be independent and the simulation will run faster.
`HeavyTailedResponse`	If this is true, Y residuals will be generated to have much heavier tails (more unusually high or low values) then a normal distribution would have.

heteroskedastic
: Whether the error variance should be allowed to depend on one of the predictor variables.

SignalStrength
: A constant used in the simulation to increase or decrease the signal-to-noise ratio; it was set to 0.5, 1, or 2 for weaker, medium or stronger signal.

Value

A list with following components: X Matrix of predictors to be screened. It will have n rows and p columns. Y Vector of responses. It will have length n.

References

Zhu, L.-P., Li, L., Li, R., & Zhu, L.-X. (2011). Model-free feature screening for ultrahigh-dimensional data. Journal of the American Statistical Association, 106, 1464-1475. <DOI:10.1198/jasa.2011.tm10563>

Examples

```
set.seed(12345678)
results <- simulateSIRS()
```

---

simulateVCM *Simulate a dataset for testing the performance of screenVCM*

Description

Simulates a dataset that can be used to test the screenVCM function, and to test the performance of the proposed method under different scenarios. The simulated dataset has a single U-covariate and p X-predictors, only a few of which have nonzero effect.

Jingyuan Liu for providing some of the code upon which this function is based.

Usage

```
simulateVCM(
  n = 200,
  rho = 0.4,
  p = 1000,
  trueIdx = c(2, 100, 400, 600, 1000),
  betaFun = NULL
)
```

Arguments

n
: Number of subjects in the simulated dataset

rho
: The correlation matrix of columns of X.

p
: The total number of features to be screened from

trueIdx	The indexes for the active features in the simulated X matrix. This should be a vector, and the values should be a subset of 1:p.
betaFun	A list of functions of U, one function for each entry in trueIdx, giving the varying effects of each active predictor in the simulated X matrix.

Value

A list with following components: X Matrix of predictors to be screened. It will have n rows and p columns. Y Vector of responses. It will have length of n. U A vector representing a covariate with which the coefficient functions vary.

Examples

```
set.seed(12345678)
results <- simulateVCM(p=1000)
```

Index