Package 'hdcrt'

May 2, 2024

Type Package

Title Hypothesis testing in high-dimensional censored-transformation models

Version 0.1.0

Author Xiao Zhang [aut,cre],
Xiangyong Tan [aut],
Runze Li [aut],
Xu Liu [aut]

Maintainer Xiao Zhang <zhangxiao1994@cuhk.edu.cn>

Description We provide an efficient censored rank-based test statistic for hypothesis testing in high-dimensional censored-transformation models. Both global test and partial test are supported.

License GPL (>= 2)

Imports Matrix

Repository github

Encoding UTF-8

LazyData true

LazyDataCompression xz

URL https://github.com/XiaoZhangryy/hscrt

BugReports https://github.com/XiaoZhangryy/hscrt/issues

RoxygenNote 7.3.1

NeedsCompilation yes

R topics documented:

hdcrpt *High dimensional censored rank partial test*

Description

High dimensional censored rank partial test

Usage

```
hdcrpt(
  x,
  y,
  status,
  z,
  smooth = c("sigmoid", "gaussian"),
  h = NULL,
  covariate_dependence = TRUE
)
```

Arguments

`x`	The design matrix.
`y`	The survival outcome.
`status`	The right censoring indicator.
`z`	The control vector, which is the control factors multiplied by the estimated coefficients.
`smooth`	The smooth function used. "sigmoid" represents the sigmoid prime kernel function and "gaussian" represents the gaussian kernel function.
`h`	The bandwidth of the control vector. The default value is the standard deviation of z divided by the square root of sample size n.
`covariate_dependence`	An indicator of whether the censoring mechanism is dependent on covariates.

Value

A list.

- ts - Test statistic.
- pval - p value.

See Also

`hdcrt`, `hdcrt_dc`

Examples

```
set.seed(0)
n <- 150
p <- 550
x <- matrix(rnorm(n * p), n, p)
u <- matrix(rnorm(n * p), n, p)
alpha <- c(rep(1, 5), rep(0, p - 5))
beta <- c(rep(1, 5), rep(0, p - 5))
y <-  u %*% alpha + x %*% beta + rnorm(n)
status <- sample(c(0, 1), n, replace = TRUE, prob = c(0.2, 0.8))
fit <- sprfabs(y, u, status)
alphahat <- fit$opttheta
z <- u %*% alphahat
test_result_sigmoid <- hdcrpt(x, y, status, z, "sigmoid")
print(test_result_sigmoid)
test_result_gaussian <- hdcrpt(x, y, status, z, "gaussian")
print(test_result_gaussian)
```

`hdcrt` *High dimensional censored rank test*

Description

High dimensional censored rank test

Usage

```
hdcrt(x, y, status, covariate_dependence = TRUE)
```

Arguments

`x`	The design matrix.
`y`	The survival outcome.
`status`	The right censoring indicator.
`covariate_dependence`	An indicator of whether the censoring mechanism is dependent on covariates.

Value

A list.

- ts - Test statistic.
- pval - p value.

See Also

`hdcrpt`, `hdcrt_dc`

Examples

```
set.seed(0)
n <- 150
p <- 550
x <- matrix(rnorm(n * p), n, p)
beta <- c(rep(1, 5), rep(0, p - 5))
y <-  x %*% beta + rnorm(n)
status <- sample(c(0, 1), n, replace = TRUE, prob = c(0.2, 0.8))
test_result <- hdcrt(x, y, status)
print(test_result)
```

`hdcrt_dc` *High dimensional censored rank test with double censored*

Description

High dimensional censored rank test with double censored

Usage

```
hdcrt_dc(x, y, status_left, status_right)
```

Arguments

`x`	The design matrix.
`y`	The survival outcome.
`status_left`	The left censoring indicator.
`status_right`	The right censoring indicator.

Value

A list.

- ts - Test statistic.
- pval - p value.

See Also

`hdcrpt`, `hdcrt`

Examples

```
set.seed(0)
n <- 150
p <- 550
x <- matrix(rnorm(n * p), n, p)
beta <- c(rep(1, 5), rep(0, p - 5))
y <-  x %*% beta + rnorm(n)
status_left <- y >= qnorm(0.1) * sqrt(6)
status_right <- y <= qnorm(0.9) * sqrt(6)
test_result <- hdcrt_dc(x, y, status_left, status_right)
print(test_result)
```

sprfabs — *A forward and backward stagewise algorithm for high-dimensional spr problem.*

Description

A forward and backward stagewise algorithm for high-dimensional spr problem.

Usage

```
sprfabs(
  y,
  x,
  status,
  eps = 0.01,
  xi = 1e-10,
  maxIter = 3000,
  sigma = NULL,
  weight = NULL,
  nmax = NULL,
  lam_m = NULL,
  message = TRUE
)
```

Arguments

`y`	The response.
`x`	The design matrix.
`status`	The right censoring indicator.
`eps`	The step size for updating coefficients. Default is `eps = 0.01`.
`xi`	The threshold for qfabs. Default is `xi = 1e-10`.
`maxIter`	The maximum number of outer-loop iterations allowed. Default is `maxIter = 3000`.
`sigma`	The tuning parameter in the Sigmoid function. Default is `NULL`.
`weight`	An optional weights. Default is 1 for each observation.
`nmax`	Limit the maximum number of variables in the model. When exceed this limit, program will early stopped. Default is `NULL`.
`lam_m`	The ratio of the minumum lambda and the maximum lambda. Default is `NULL`.
`message`	An indicator of whether print warning messages.

Value

A list.

- theta - The estimation of covariates.
- opttheta - The optimal estimation of covariates.
- lambda - lambda sequence.

- direction - direction sequence.
- iter - Iterations.
- bic - The EBIC for each solution.
- loss - loss for each solution.
- df - Number of nonzero coefficients.
- opt - Position of the optimal lambda based on EBIC.

Examples

```
set.seed(0)
n <- 150
p <- 550
x <- matrix(rnorm(n * p), n, p)
alpha <- c(rep(1, 5), rep(0, p - 5))
y <-  x %*% alpha + rnorm(n)
status <- sample(c(0, 1), n, replace = TRUE, prob = c(0.2, 0.8))
fit <- sprfabs(y, x, status)
alphahat <- as.vector(fit$opttheta)
print(which(alphahat != 0))
print(alphahat[alphahat != 0])
```

Index