Package 'MTAFT'

November 13, 2023

Type Package

Title Data-Driven Estimation for Multi-Threshold Accelerate Failure
Time Model

Version 0.1.0

Date 2023-11-10

Maintainer Chuang WAN <wanchuang@nankai.edu.cn>

Description Developed a data-driven estimation framework for the multi-threshold accelerate failure time (MTAFT) model. The MTAFT model features different linear forms in different subdomains, and one of the major challenges is determining the number of threshold effects. The package introduces a data-driven approach that utilizes a Schwarz' information criterion, which demonstrates consistency under mild conditions. Additionally, a cross-validation (CV) criterion with an order-preserved sample-splitting scheme is proposed to achieve consistent estimation, without the need for additional parameters. The package establishes the asymptotic properties of the parameter estimates and includes an efficient score-type test to examine the existence of threshold effects. The methodologies are supported by numerical experiments and theoretical results, showcasing their reliable performance in finite-sample cases.

License GPL-3

Depends R (>= 3.5.0)

Suggests MASS, knitr, rmarkdown

RoxygenNote 7.2.3

Encoding UTF-8

Imports graphics, methods, stats, grpreg

NeedsCompilation no

Author Chuang WAN [aut, cre],
Hao ZENG [aut],
Wei ZHONG [aut],
Changliang ZOU [aut]

Repository CRAN

Date/Publication 2023-11-13 17:43:21 UTC

R topics documented:

MTAFT_CV   *MTAFT_CV: Cross-Validation for Multiple Thresholds Accelerated Failure Time Model*

Description

This function implements a cross-validation method for the multiple thresholds accelerated failure time (AFT) model using either the "WBS" (Wild Binary Segmentation) or "DP" (Dynamic Programming) algorithm. It determines the optimal number of thresholds by evaluating the cross-validation (CV) values.

Usage

```
MTAFT_CV(
  Y,
  X,
  delta,
  Tq,
  algorithm,
  dist_min = 50,
  ncps_max = 4,
  wbs_nintervals = 200
)
```

Arguments

Y	the censored logarithm of the failure time.
X	the design matrix without the intercept.
delta	the censoring indicator.
Tq	the threshold values.
algorithm	the threshold detection algorithm, either "WBS" or "DP".
dist_min	the pre-specified minimal number of observations within each subgroup. Default is 50.
ncps_max	the pre-specified maximum number of thresholds. Default is 4.
wbs_nintervals	the number of random intervals in the WBS algorithm. Default is 200.

Value

A list with the following components:

params the subgroup-specific slope estimates and variance estimates.

thres the threshold estimates.

CV_vals the CV values for all candidate number of thresholds.

Examples

```
# Generate simulated data with 500 samples and normal error distribution
dataset <- MTAFT_simdata(n = 500, err = "normal")

Y <- dataset[, 1]
delta <- dataset[, 2]
Tq <- dataset[, 3]
X <- dataset[, -c(1:3)]

# Run mAFT_CV with WBS algorithm
maft_cv_result <- MTAFT_CV(Y, X, delta, Tq, algorithm = "WBS")
maft_cv_result$params
maft_cv_result$thres
maft_cv_result$CV_vals
```

MTAFT_IC   *MTAFT_IC: Multiple Thresholds Accelerated Failure Time Model with Information Criteria*

Description

This function implements a method for multiple thresholds accelerated failure time (AFT) model with information criteria. It estimates the subgroup-specific slope coefficients and variance estimates, as well as the threshold estimates using either the "WBS" (Wild Binary Segmentation) or "DP" (Dynamic Programming) algorithm.

Usage

```
MTAFT_IC(
  Y,
  X,
  delta,
  Tq,
  c0 = 0.299,
  delta0 = 2.01,
  algorithm = c("WBS", "DP"),
  dist_min = 50,
  ncps_max = 4,
  wbs_nintervals = 200
)
```

Arguments

Y	the censored logarithm of the failure time.
X	the design matrix without the intercept.
delta	the censoring indicator.
Tq	the threshold values.
c0	the penalty factor c0 in the information criteria (IC), default is 0.299.
delta0	the penalty factor delta0 in the information criteria (IC), default is 2.01.
algorithm	the threshold detection algorithm, either "WBS" or "DP". Default is "WBS".
dist_min	the pre-specified minimal number of observations within each subgroup. Default is 50.
ncps_max	the pre-specified maximum number of thresholds. Default is 4.
wbs_nintervals	the number of random intervals in the WBS algorithm. Default is 200.

Value

A list with the following components:

params the subgroup-specific slope estimates and variance estimates.

thres the threshold estimates.

IC_val the IC values for all candidate number of thresholds.

References

(Add relevant references here)

Examples

```
# Generate simulated data with 500 samples and normal error distribution
dataset <- MTAFT_simdata(n = 500, err = "normal")
Y <- dataset[, 1]
delta <- dataset[, 2]
Tq <- dataset[, 3]
X <- dataset[, -c(1:3)]

# Run MTAFT_IC with WBS algorithm
mtaft_ic_result <- MTAFT_IC(Y, X, delta, Tq, algorithm = 'WBS')
mtaft_ic_result$params
mtaft_ic_result$thres
mtaft_ic_result$IC_val
```

MTAFT_simdata   *Generate simulated data for MTAFT analysis.*

Description

This function generates simulated data for the MTAFT (Multi-Threshold Accelerated Failure Time) analysis based on a simple simulation procedure described in the article.

Usage

```
MTAFT_simdata(n, err = c("normal", "t3"))
```

Arguments

n	The number of sample size.
err	The error distribution type, either "normal" or "t3".

Value

A dataset containing the simulated data for MTAFT analysis.

Examples

```
# Generate simulated data with 500 samples and normal error distribution
dataset <- MTAFT_simdata(n = 500, err = "normal")
Y <- dataset[, 1]
delta <- dataset[, 2]
Tq <- dataset[, 3]
X <- dataset[, -c(1:3)]

# Generate simulated data with 200 samples and t3 error distribution
dataset <- MTAFT_simdata(n = 200, err = "t3")
Y <- dataset[, 1]
delta <- dataset[, 2]
Tq <- dataset[, 3]
X <- dataset[, -c(1:3)]
```

MTAFT_test   *Perform score-type test for the presence of threshold effect in multi-threshold situations.*

Description

This function performs a score-type test statistics for the presence of threshold effect in multi-threshold situations.

Usage

```
MTAFT_test(Y, X, Tq, delta, nboots)
```

Arguments

Y	Response variable.
X	Covariates.
Tq	Threshold variable.
delta	Indicator vector for censoring.
nboots	Number of bootstrap iterations.

Value

p-value result indicating the presence of threshold effect.

Examples

```
# Generate simulated data with 500 samples and normal error distribution
dataset <- MTAFT_simdata(n = 500, err = "normal")
Y <- dataset[, 1]
delta <- dataset[, 2]
Tq <- dataset[, 3]
X <- dataset[, -c(1:3)]

# Perform score-type test with 500 bootstraps
pval <- MTAFT_test(Y, X, Tq, delta, nboots = 500)

# Perform score-type test with 1000 bootstraps
pval <- MTAFT_test(Y, X, Tq, delta, nboots = 1000)
```

TSMCP	*TSMCP: Two stage multiple change points detection for AFT model.*

Description

This function first formulates the threshold problem as a group model selection problem so that a concave 2-norm group selection method can be applied using the 'grpreg' package in R, and then finalizes it via a refining method.

Usage

```
TSMCP(Y, X, delta, c, penalty = "scad")
```

Arguments

Y	the censored logarithm of the failure time.
X	the design matrix without the intercept.
delta	the censoring indicator.
c	the length of each segment in the splitting stage, defined as `ceiling(c * sqrt(length(Y)))`.
penalty	Penalty type (default is "scad").

Value

An object with the following components:

cp the change points.

coef the estimated coefficients.

sigma the variance of the error.

residuals the residuals.

Yn weighted Y by Kaplan-Meier weight.

Xn weighted Xn by Kaplan-Meier weight.

References

Li, Jialiang, and Baisuo Jin. 2018. "Multi-Threshold Accelerated Failure Time Model." The Annals of Statistics 46 (6A): 2657–82.

See Also

grpreg

Examples

```
library(grpreg)
# Generate simulated data with 500 samples and normal error distribution
dataset <- MTAFT_simdata(n = 500, err = "normal")
Y <- dataset[, 1]
delta <- dataset[, 2]
Tq <- dataset[, 3]
X <- dataset[, -c(1:3)]
n1 = sum(delta)
c=seq(0.5,1.5,0.1)
m=ceiling(c*sqrt(n1))
bicy= rep(NA,length(c))
tsmc=NULL
p = ncol(X)
for(i in 1:length(c)){
    tsm=try(TSMCP(Y,X,delta,c[i],penalty = "scad"),silent=TRUE)
    if(is(tsm,"try-error")) next()
    bicy[i]=log(n)*((length(tsm[[1]])+1)*(p+1))+n*log(tsm[[3]])
    tsmc[[i]]=tsm
```

```
    }

    if((any(!is.na(bicy)))){
        tsmcp=tsmc[[which(bicy==min(bicy))[1]]]
        thre.LJ = Tq[tsmcp[[1]]]
        thre.num.Lj = length(thre.LJ)
        thre.LJ
        thre.num.Lj
    }
```

Index