Package ‘hds’

December 31, 2016

Type    Package
Version  0.8.1
Title    Hazard Discrimination Summary
Description Functions for calculating the hazard discrimination summary and its standard errors, as described in Liang and Heagerty (2016) <doi:10.1111/biom.12628>.
Date     2016-12-30
URL      https://github.com/liangcj/hds
BugReports https://github.com/liangcj/hds/issues
Imports  stats, survival, tensor
License  GPL-2
LazyData TRUE
Depends  R (>= 3.1.0)
RoxygenNote  5.0.1
NeedsCompilation no
Author    C. Jason Liang [aut, cre]
Maintainer C. Jason Liang <liangcj@gmail.com>
Repository CRAN
Date/Publication 2016-12-31 16:32:12

R topics documented:

finda ....................................................... 2
hds .......................................................... 3
hdsle ....................................................... 4
hdsle.fast .................................................. 6
hdslese.fast ............................................. 7
hds_l ........................................................ 8
pbc5 ......................................................... 9

Index 10
finda

Estimate the time-varying coefficients from a local-in-time Cox model

Description
finda estimates the time-varying coefficients beta(t) at a single time from a local-in-time Cox model. Think of it as a Cox model where the the coefficients are allowed to vary with time. Further details can be found in Cai and Sun (2003) and Tian et al. (2005).

Usage
finda(tt, times, status, covars, start = rep(0, ncol(covars)), h = 400, ...)

Arguments
- tt: Time to estimate beta(t) at
- times: A vector of observed follow up times.
- status: A vector of status indicators, usually 0=alive, 1=dead.
- covars: A matrix or data frame of numeric covariate values, with a column for each covariate and each observation is on a separate row.
- start: A vector of length p of starting values to be passed to optim for the numerical optimization procedure. p is the number of covariates. Defaults to all zeroes.
- h: A single value on the time scale representing the bandwidth to use.
- ...: Additional parameters to pass to optim.

Details
The naming of the function finda stands for "find a(t)", where "a(t)" is the notation used in Cai and Sun (2003) to represent the time-varying Cox model coefficients. We also refer to "a(t)" as "beta(t)" through the documentation.

The user typically will not interact with this function, as finda is wrapped by hdslc.

Value
A vector of length p, where p is the number of covariates. The vector is the estimated beta(t) from the local-in-time Cox model at time tt.

References
Description

Returns hazard discrimination summary (HDS) estimates at all specified evaluation times. See Liang and Heagerty (2016) for details on HDS.

Usage

hds(times, status, m, evaltimes = times[order(times)], se = TRUE)

Arguments

times A vector of observed follow up times.
status A vector of status indicators, usually 0=alive, 1=dead.
m A matrix or data frame of covariate values, with a column for each covariate and each observation is on a separate row. Non-numeric values are acceptable, as the values will be transformed into a numeric model matrix through survival::coxph.
evaltimes A vector of times at which to estimate HDS. Defaults to all the times specified by the times vector. If there are a lot of observations, then you may want to enter in a sparser vector of times for faster computation.
se TRUE or FALSE. TRUE: calculate and return standard error estimates. FALSE: do not calculate standard errors estimates and return NAs. Defaults to TRUE. May want to set to FALSE to save computation time if using this function to compute bootstrap standard errors.

Details

A wrapper for hds_t. Since hds_t only estimates HDS at one time point, this function calls hds_t multiple times to estimate the entire HDS curve. hds and hds1c are the main functions the user will interact with in this package.

The covariate values m are centered for numerical stability. This is particularly relevant for the standard error calculations.

Value

A data frame with three columns: 1) the evaluation times, 2) the HDS estimates at each evaluation time, and 3) the standard error estimates at each evaluation time

References

See Also

hdslc

Examples

```r
## Not run:
head(hds(times = survival::pbc[1:312, 2],
          status = (survival::pbc[1:312, 3] == 2) * 1,
          m = survival::pbc[1:312, 5]))

hdsres <- hds(times = pbc5[, 1], status = pbc5[, 2], m = pbc5[, 3:7])
hdslcres <- hdslc(times = pbc5[, 1], status = pbc5[, 2], m = pbc5[, 3:7], h = 730)
Surv <- summary(survival::survfit(survival::Surv(pbc5[, 1], pbc5[, 2]) ~ 1))
Survtd <- cbind(Surv$time, c(0, diff(1 - Surv$surv)))
tden <- density(x = Survtd[, 1], weights = Survtd[, 2], bw = 100, kernel = "epanechnikov")

par(mar = c(2.25, 2.25, 0, 0) + 0.1, mgp = c(1.25, 0.5, 0))
plot(c(hdslcres[, 1], hdslcres[, 1]), c(hdslcres[, 2] - 1.96 * hdslcres[, 3],
                                          hdslcres[, 2] + 1.96 * hdslcres[, 3]),
     type = "n", xlab = "days", ylab = "HDS(t)", cex.lab = .75, cex.axis = .75,
     ylim = c(-3, 15), xlim = c(0, 3650))
polygon(x = c(hdsres[, 1], hdsres[312:1, 1]), col = rgb(1, 0, 0, .25), border = NA,
         fillOddEven = TRUE, y = c(hdsres[, 2] + 1.96 * hdsres[, 3],
                                   (hdsres[, 2] - 1.96 * hdsres[, 3])[-312:1]))
polygon(x = c(hdslcres[, 1], hdslcres[312:1, 1]), col = rgb(0, 0, 1, .25), border = NA,
         fillOddEven = TRUE, y = c(hdslcres[, 2] + 1.96 * hdslcres[, 3],
                                   (hdslcres[, 2] - 1.96 * hdslcres[, 3])[-312:1]))
lines(hdsres[, 1], hdsres[, 2], lwd = 2, col = "red")
lines(hdslcres[, 1], hdslcres[, 2], lwd = 2, col = "blue")
abline(h = 1, lty = 3)
legend(x = 1200, y = 14, legend = c("Proportional hazards",
                                       "Local-in-time proportional hazards",
                                       "Time density"), col = c("red", "blue", "gray"),
                                       lwd = 2, bty = "n", cex = .75)
with(tden, polygon(c(x, x[length(x)]:1),
                   c(y * 3 / max(y) - 3.5, rep(-3.5, length(x))),
                   col = "gray", border = NA, fillOddEven = TRUE))

## End(Not run)
```

hdslc  

**Hazard discrimination summary estimator**

Description

Returns local constant HDS estimates at all specified evaluation times. See Liang and Heagerty (2016) for details on HDS.
Usage

```r
hdslc(times, status, m, evaltimes = times[order(times)], h = 1.06 * 
       sd(times) * (length(times)^(-0.2)), se = TRUE)
```

Arguments

- `times`: A vector of observed follow up times.
- `status`: A vector of status indicators, usually 0=alive, 1=dead.
- `m`: A matrix or data frame of covariate values, with a column for each covariate and each observation on a separate row. Non-numeric values are acceptable, as the values will be transformed into a numeric model matrix through `survival::coxph`.
- `evaltimes`: A vector of times at which to estimate HDS. Defaults to all the times specified by the `times` vector. If there are a lot of observations, then you may want to enter in a sparser vector of times for faster computation.
- `h`: A single numeric value representing the bandwidth to use, on the time scale. The default bandwidth is a very ad hoc estimate using Silverman’s rule of thumb.
- `se`: TRUE or FALSE. TRUE: calculate and return standard error estimates. FALSE: do not calculate standard errors estimates and return NAs. Defaults to TRUE. May want to set to FALSE to save computation time if using this function to compute bootstrap standard errors.

Details

A local constant version of `hds`. While `hds` estimates HDS(t) assuming the Cox proportional hazards model, `hdslc` estimates HDS(t) using a relaxed, local-in-time Cox model. Specifically, the hazard ratios are allowed to vary with time. See Cai and Sun (2003) and Tian Zucker Wei (2005) for details on the local-in-time Cox model.

Point estimates use `hdslc.fast` and standard errors use `hdslcse.fast`. `hdslc.fast` requires an estimate of beta(t) (time-varying hazard ratio), which is estimated using `finda()`; and subject specific survival, which is estimated using `sssf.fast()`. `hdslcse.fast` requires the same and in addition standard error estimates of beta(t), which are estimated using `betahatse.fast()`.

The covariate values `m` are centered for numerical stability. This is particularly relevant for the standard error calculations.

Value

A data frame with three columns: 1) the evaluation times, 2) the HDS estimates at each evaluation time, and 3) the standard error estimates at each evaluation time.

References


See Also

`hds`, `finda`

Examples

```r
## not run:
head(hdslc(times = survival::pbc[1:312, 2],
      status = (survival::pbc[1:312, 3]==2)*1,
      m = survival::pbc[1:312, 5]))

hdslcres <- hdslc(times = pbc[, 1], status = pbc[, 2], m = pbc[, 3:7], h = 730)
Surv <- summary(survival::survfit(survival::Surv(pbc[, 1], pbc[, 2])-1))
Survtd <- cbind(Surv$time, c(0, diff(1-Surv$surv)))
tden <- density(x=survtd[, 1], weights=survtd[, 2], bw=100, kernel="epanechnikov")

par(mar=c(2.25, 2.25, 0, 0)+0.1, mgp=c(1.25, 0.5, 0))
plot(c(hdslcres[, 1], hdslcres[, 1]), c(hdslcres[, 2] - 1.96*hdslcres[, 3],
     hdslcres[, 2] + 1.96*hdslcres[, 3]),
     type="n", xlab="days", ylab="HDS(t)", cex.lab=.75, cex.axis=.75,
     ylim=c(-3.15), xlim=c(0,3650))
polygon(x=c(hdslcres[, 1], hdslcres[312:1, 1]), col=rgb(1,0,0,.25), border=NA,
        fillOddEven=TRUE, y=c(hdslcres[, 2]+1.96*hdslcres[, 3],
        (hdslcres[,2]-1.96*hdslcres[,3])[312:1]))
polygon(x=c(hdslcres[, 1], hdslcres[312:1, 1]), col=rgb(0,0,1,.25), border=NA,
        fillOddEven=TRUE, y=c(hdslcres[, 2] + 1.96*hdslcres[, 3],
        (hdslcres[,2] - 1.96*hdslcres[,3])[312:1]))
lines(hdslcres[, 1], hdslcres[, 2], lwd=2, col="red")
lines(hdslcres[, 1], hdslcres[, 2], lwd=2, col="blue")
abline(h=1, lty=3)
legend(x=1200, y=14, legend=c("Proportional hazards",
   "Local-in-time proportional hazards",
   "Time density"), col=c("red", "blue", "gray"),
  lwd=2, bty="n", cex=0.75)
with(tden, polygon(c(x, x[length(x):1]),
  c(y*3/max(y)-3.5, rep(-3.5, length(x))),
  col="gray", border=NA, fillOddEven=TRUE))

## End(Not run)
```
Description


Usage

`hdslc.fast(S, betahat, m)`

Arguments

- `S`: A vector of length `nrow(m)` (which is typically the number of observations n), where each value is the subject-specific survival at time t where t is implied by the choice of `betahat`.
- `betahat`: A p x 1 vector of coefficient estimates at time t of interest from the local-in-time Cox model. Vector length p is the number of covariates. Typically the output from `hdslc::finda` is passed here.
- `m`: A numeric n x p matrix of covariate values, with a column for each covariate and each observation is on a separate row.

Details

The user typically will not interact with this function. Rather, `hdslc` wraps this function and is what the user typically will use.

Value

The HDS estimate at times t, where t is implied by choice of S and `betahat` passed to `hdslc.fast`.

---

Description

`hdslcse.fast` calculates an estimate of the variance for the local constant hazard discrimination summary estimator at a time t. The time t is implied by S, `betahat`, and `betahatse`.

Usage

`hdslcse.fast(S, betahat, m, betahatse)`
Arguments

- \( S \): A vector of length \( nrow(m) \) (which is typically the number of observations \( n \)), where each value is the subject-specific survival at time \( t \) where \( t \) is implied by the choice of \( \text{betahat} \).

- \( \text{betahat} \): A \( p \times 1 \) vector of coefficient estimates at time \( t \) of interest from the local-in-time Cox model. Vector length \( p \) is the number of covariates. Typically the output from \text{hdslc}::\text{finda} is passed here.

- \( m \): A numeric \( n \times p \) matrix of covariate values, with a column for each covariate and each observation is on a separate row.

- \( \text{betahatse} \): A \( p \times p \) covariance matrix for \( \text{betahat} \) at time \( t \)

Details

The use will typically not interact with this function directly. Instead this function is wrapped by \text{hdslc}.

Value

Variance estimate that has not been normalized. To get a usable standard error estimate, divide the output of this function by the bandwidth and sample size, and then take the square root.

\[ hds_t \]

\textit{Hazard discrimination summary estimate at one time point}

Description

\( hds_t \) estimates HDS at time \( t \) under the PH assumption

Usage

\( hds_t(t, L0hat, \text{betahat}, m) \)

Arguments

- \( t \): The time at which to calculate HDS

- \( L0hat \): A data frame with variable names of hazard and time. Typically the object returned by \text{basehaz}.

- \( \text{betahat} \): A vector of coefficient estimates from the Cox model. Typically the \text{coefficients} value from the \text{coxph} object object returned by \text{coxph}.

- \( m \): A numeric matrix of covariate values, with a column for each covariate and each observation is on a separate row.

Details

The user typically will not interact with this function. Rather, \( hds \) is a wrapper for this function and is what the user typically will use.
Description

A cleaned up version of the Mayo PBC data from \texttt{survival::pbc}. Only the first 312 observations are used (the cases who participated in the randomized trial). Only five of the covariates (listed below) are used. Further, two of the covariates were log transformed.

Usage

\texttt{pbc5}

Format

A data frame with 312 rows and 7 variables:

- \texttt{time}  follow up time in days
- \texttt{status}  1=death, 0=censored
- \texttt{age}  age in years
- \texttt{edema}  0=no edema, 0.5=untreated or successfully treated, 1=edema despite diuretic therapy
- \texttt{bili}  log serum bilirubin level (original value from \texttt{survival::pbc} is unlogged)
- \texttt{albumin}  serum albumin
- \texttt{protime}  log standardized blood clotting time (original value from \texttt{survival::pbc} is unlogged)

Source

Cleaned up version of \texttt{survival::pbc}
Index

*Topic datasets
  pbc5, 9

finda, 2, 6

hds, 3, 6
hds_t, 8
hdslc, 4, 4
hdslc.fast, 6
hdslcse.fast, 7

pbc5, 9