Package 'causalCmprsk'

August 9, 2021

Type Package

Title Nonparametric and Cox-Based Estimation of ATE in Competing Risks

Version 1.0.3

Author Bella Vakulenko-Lagun, Colin Magdamo, Marie-Laure Charpignon, Bang Zheng, Mark Albers, Sudeshna Das

Maintainer Bella Vakulenko-Lagun <blagun@stat.haifa.ac.il>

Description Estimation of average treatment effects (ATE) of two static treatment regimes on time-to-event outcomes with K competing events (K can be 1). The method uses propensity scores weighting for emulation of baseline randomization.

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 3.6)

Imports survival, inline, doParallel, parallel, utils, foreach, data.table, purrr

RoxygenNote 7.1.1

Suggests knitr, rmarkdown, bookdown, tidyverse, ggalt, cobalt, ggsci, modEvA, naniar, DT, Hmisc, hrbrthemes, summarytools, kableExtra

VignetteBuilder knitr

URL https://github.com/Bella2001/causalCmprsk

BugReports https://github.com/Bella2001/causalCmprsk/issues

NeedsCompilation no

Repository CRAN

Date/Publication 2021-08-09 14:50:02 UTC

R topics documented:

`causalCmprsk` *Estimation of Average Treatment Effects (ATE) on Time-to-Event Outcomes with Competing Events*

Description

The package accompanies the paper of Charpignon et al. (2021). It can be applied to data with any number of competing events, including the case of only one type of event. The method uses propensity scores weighting for emulation of baseline randomization. The package implements different types of weights: ATE, stabilized ATE, ATT, ATC and overlap weights, as described in Li et al. (2018), and different treatment effect measures (hazard ratios, risk differences, risk ratios, and restricted mean time differences).

Details

The **causalCmprsk** package provides two main functions: `fit.cox` for Cox-based estimation and `fit.nonpar` that does not assume any model for potential outcomes. The function `get.weights` returns estimated weights that are aimed for emulation of a baseline randomization in observational data where the treatment was not assigned randomly. The function `get.pointEst` extracts a point estimate corresponding to a specific time point from the time-varying functionals returned by `fit.cox` and `fit.nonpar`. The function `get.numAtRisk` allows to obtain the number-at-risk statistic in the raw and weighted data.

References

A.F. Connors, T. Speroff, N.V. Dawson, C. Thomas, F.E. Harrell, D. Wagner, N. Desbiens, et al. 1996. The Effectiveness of Right Heart Catheterization in the Initial Care of Critically Ill Patients. Journal of the American Medical Association 276: 889–97.

M.-L. Charpignon, B. Vakulenko-Lagun, B. Zheng, C. Magdamo, B. Su, K.E. Evans, S. Rodriguez, et al. 2021. Metformin's relationship with dementia and survival in emulated trials of real-world patient data and in systems pharmacology. submitted.

F. Li, K.L. Morgan, and A.M. Zaslavsky. 2018. Balancing Covariates via Propensity Score Weighting. Journal of the American Statistical Association 113 (521): 390–400.

`fit.cox` *Cox-based estimation of ATE corresponding to the target population*

Description

Implements Cox-based estimation of ATE assuming a structural proportional hazards model for two potential outcomes. It provides three measures of treatment effects on time-to-event outcomes: (1) cause-specific hazard ratios which are time-dependent measures under a nonparametric model, (2) risk-based measures such as cause-specific risk differences and cause-specific risk ratios, and (3) restricted-mean-time differences which quantify how much time on average was lost (or gained) due to treatment by some specified time point. Please see our package vignette for more details.

Usage

```
fit.cox(
  df,
  X,
  E,
  A,
  C = NULL,
  wtype = "unadj",
  cens = 0,
  conf.level = 0.95,
  bs = FALSE,
  nbs.rep = 400,
  seed = 17,
  parallel = FALSE,
  verbose = FALSE
)
```

Arguments

`df`	a data frame that includes time-to-event `X`, type of event `E`, a treatment indicator `A` and covariates `C`.
`X`	a character string specifying the name of the time-to-event variable in `df`.
`E`	a character string specifying the name of the "event type" variable in `df`.
`A`	a character specifying the name of the treatment/exposure variable. It is assumed that `A` is a numeric binary indicator with 0/1 values, where `A`=1 is assumed a treatment group, and `A`=0 a control group.
`C`	a vector of character strings with variable names (potential confounders) in the logistic regression model for Propensity Scores, i.e. P(A=1|C=c). The default value of `C` is NULL corresponding to `wtype`="unadj" that will estimate treatment effects in the raw (observed) data.
`wtype`	a character string variable indicating the type of weights that will define the target population for which the ATE will be estimated. The default is "unadj" -

	this will not adjust for possible treatment selection bias and will not use propensity scores weighting. It can be used, for example, in data from a randized controlled trial (RCT) where there is no need for emulation of baseline randomization. Other possible values are "stab.ATE", "ATE", "ATT", "ATC" and "overlap". See Table 1 from Li, Morgan, and Zaslavsky (2018). "stab.ATE" is defined as P(A=a)/P(A=a|C=c) - see Hernán et al. (2000).
`cens`	an integer value in `E` that corresponds to censoring times recorded in `X`. By default `fit.nonpar` assumes `cens=0`
`conf.level`	the confidence level that will be used in the bootstrap confidence intervals. The default is 0.95
`bs`	a logical flag indicating whether to perform bootstrap in order to obtain confidence intervals. There are no analytical confidence intervals in `fit.nonpar`
`nbs.rep`	number of bootstrap replications
`seed`	the random seed for the bootstrap, in order to make the results reproducible
`parallel`	a logical flag indicating whether to perform bootstrap sequentially or in parallel using 2 cores simultaneously. The default value is FALSE
`verbose`	a logical flag indicating whether to show a progress of bootstrap. The progress bar is shown only for sequential bootstrap computation. The default value is FALSE.

Value

A list with the following fields:

`time`
a vector of time points for which all the parameters are estimated
`trt.0`
a list of estimates of the absolute counterfactual parameters corresponding to `A=0` and the type of event `E`. `trt.0` has the numb

- `CumHaz` a vector of cumulative hazard estimates
- `CIF` a vector of cumulative incidence functions
- `RMT` a vector of restricted mean time estimates

`trt.1`
a list of estimates of the absolute counterfactual parameters corresponding to `A=1` and the type of event `E`. `trt.1` has the numb

- `CumHaz` a vector of cumulative hazard estimates
- `CIF` a vector of cumulative incidence functions
- `RMT` a vector of restricted mean time estimates

`trt.eff`
a list of estimates of the treatment effect measures corresponding to the type of event `E`. `trt.eff` has the number of fields as th

- `log.CumHazR` an estimate of the log of the hazard ratio. It is a scalar since the Cox model is assumed.
- `RD` a vector of time-varying risk difference between two treatment arms
- `RR` a vector of time-varying risk ratio between two treatment arms
- `ATE.RMT` a vector of the time-varying restricted mean time difference between two treatment arms

References

F. Li, K.L. Morgan, and A.M. Zaslavsky. 2018. Balancing Covariates via Propensity Score Weighting. Journal of the American Statistical Association, 113 (521): 390–400.

M.A. Hernán, B. Brumback, and J.M. Robins. 2000. Marginal structural models and to estimate the causal effect of zidovudine on the survival of HIV-positive men. Epidemiology, 11 (5): 561-570.

See Also

`fit.nonpar`, `get.pointEst`, `causalCmprsk`

Examples

```
# create a data set
n <- 1000
set.seed(7)
c1 <- runif(n)
c2 <- as.numeric(runif(n)< 0.2)
set.seed(77)
cf.m.T1 <- rweibull(n, shape=1, scale=exp(-(-1 + 2*c1)))
cf.m.T2 <-  rweibull(n, shape=1, scale=exp(-(1 + 1*c2)))
cf.m.T <- pmin( cf.m.T1, cf.m.T2)
cf.m.E <- rep(0, n)
cf.m.E[cf.m.T1<=cf.m.T2] <- 1
cf.m.E[cf.m.T2<cf.m.T1] <- 2
set.seed(77)
cf.s.T1 <- rweibull(n, shape=1, scale=exp(-1*c1 ))
cf.s.T2 <-  rweibull(n, shape=1, scale=exp(-2*c2))
cf.s.T <- pmin( cf.s.T1, cf.s.T2)
cf.s.E <- rep(0, n)
cf.s.E[cf.s.T1<=cf.s.T2] <- 1
cf.s.E[cf.s.T2<cf.s.T1] <- 2
exp.z <- exp(0.5 + 1*c1 - 1*c2)
pr <- exp.z/(1+exp.z)
TRT <- ifelse(runif(n)< pr, 1, 0)
X <- ifelse(TRT==1, cf.m.T, cf.s.T)
E <- ifelse(TRT==1, cf.m.E, cf.s.E)
covs.names <- c("c1", "c2")
data <- data.frame(X=X, E=E, TRT=TRT, c1=c1, c2=c2)
wei <- get.weights(data, "TRT", covs.names, wtype = "overlap")
hist(wei$ps[data$TRT==1], col="red", breaks = seq(0,1,0.05))
par(new=TRUE)
hist(wei$ps[data$TRT==0], col="blue", breaks = seq(0,1,0.05))
```

```
# Cox-based estimation:
res.cox.ATE <- fit.cox(df=data, X="X", E="E", A="TRT", C=covs.names, wtype="stab.ATE")
cox.pe <- get.pointEst(res.cox.ATE, 0.5)
cox.pe$trt.eff[[1]]$RD #-0.1894038

# please see our package vignette for practical examples
```

`fit.nonpar` *Nonparametric estimation of ATE corresponding to the target population*

Description

Implements nonparametric estimation (based on the weighted Aalen-Johansen estimator) of ATE meaning that it does not assume any model for potential outcomes. It provides three measures of treatment effects on time-to-event outcomes: (1) cause-specific hazard ratios which are time-dependent measures under a nonparametric model, (2) risk-based measures such as cause-specific risk differences and cause-specific risk ratios, and (3) restricted-mean-time differences which quantify how much time on average was lost (or gained) due to treatment by some specified time point. Please see our package vignette for more details.

Usage

```
fit.nonpar(
  df,
  X,
  E,
  A,
  C = NULL,
  wtype = "unadj",
  cens = 0,
  conf.level = 0.95,
  bs = FALSE,
  nbs.rep = 400,
  seed = 17,
  parallel = FALSE,
  verbose = FALSE
)
```

Arguments

`df`	a data frame that includes time-to-event `X`, type of event `E`, a treatment indicator `A` and covariates `C`.
`X`	a character string specifying the name of the time-to-event variable in `df`.
`E`	a character string specifying the name of the "event type" variable in `df`.

A	a character specifying the name of the treatment/exposure variable. It is assumed that A is a numeric binary indicator with 0/1 values, where A=1 is assumed a treatment group, and A=0 a control group.
C	a vector of character strings with variable names (potential confounders) in the logistic regression model for Propensity Scores, i.e. P(A=1|C=c). The default value of C is NULL corresponding to wtype="unadj" that will estimate treatment effects in the raw (observed) data.
wtype	a character string variable indicating the type of weights that will define the target population for which the ATE will be estimated. The default is "unadj" - this will not adjust for possible treatment selection bias and will not use propensity scores weighting. It can be used, for example, in data from a randized controlled trial (RCT) where there is no need for emulation of baseline randomization. Other possible values are "stab.ATE", "ATE", "ATT", "ATC" and "overlap". See Table 1 from Li, Morgan, and Zaslavsky (2018). "stab.ATE" is defined as P(A=a)/P(A=a|C=c) - see Hernán et al. (2000).
cens	an integer value in E that corresponds to censoring times recorded in X. By default fit.nonpar assumes cens=0
conf.level	the confidence level that will be used in the bootstrap confidence intervals. The default is 0.95
bs	a logical flag indicating whether to perform bootstrap in order to obtain confidence intervals. There are no analytical confidence intervals in fit.nonpar
nbs.rep	number of bootstrap replications
seed	the random seed for the bootstrap, in order to make the results reproducible
parallel	a logical flag indicating whether to perform bootstrap sequentially or in parallel using 2 cores simultaneously. The default value is FALSE
verbose	a logical flag indicating whether to show a progress of bootstrap. The progress bar is shown only for sequential bootstrap computation. The default value is FALSE.

Value

A list with the following fields:

time
a vector of time points for which all the parameters are estimated
trt.0
a list of estimates of the absolute counterfactual parameters corresponding to A=0 and the type of event E. trt.0 has the numb

- CumHaz a vector of cumulative hazard estimates
- CIF a vector of cumulative incidence functions
- RMT a vector of restricted mean time estimates

trt.1
a list of estimates of the absolute counterfactual parameters corresponding to A=1 and the type of event E. trt.1 has the numb

- CumHaz a vector of cumulative hazard estimates
- CIF a vector of cumulative incidence functions
- RMT a vector of restricted mean time estimates

trt.eff
a list of estimates of the treatment effect measures corresponding to the type of event E. trt.eff has the number of fields as th

- log.CumHazR a vector of the log of the time-varying ratio of hazards in two treatment arms
- RD a vector of time-varying risk difference between two treatment arms
- RR a vector of time-varying risk ratio between two treatment arms
- ATE.RMT a vector of the time-varying restricted mean time difference between two treatment arms

References

F. Li, K.L. Morgan, and A.M. Zaslavsky. 2018. Balancing Covariates via Propensity Score Weighting. Journal of the American Statistical Association 113 (521): 390–400.

M.A. Hernán, B. Brumback, and J.M. Robins. 2000. Marginal structural models and to estimate the causal effect of zidovudine on the survival of HIV-positive men. Epidemiology, 11 (5): 561-570.

See Also

fit.cox, get.pointEst, causalCmprsk

Examples

```
# create a data set
n <- 1000
set.seed(7)
c1 <-  runif(n)
c2 <- as.numeric(runif(n)< 0.2)
set.seed(77)
cf.m.T1 <- rweibull(n, shape=1, scale=exp(-(-1 + 2*c1)))
cf.m.T2 <-  rweibull(n, shape=1, scale=exp(-(1 + 1*c2)))
cf.m.T <- pmin( cf.m.T1, cf.m.T2)
cf.m.E <- rep(0, n)
cf.m.E[cf.m.T1<=cf.m.T2] <- 1
cf.m.E[cf.m.T2<cf.m.T1] <- 2
set.seed(77)
cf.s.T1 <- rweibull(n, shape=1, scale=exp(-1*c1 ))
cf.s.T2 <-  rweibull(n, shape=1, scale=exp(-2*c2))
cf.s.T <- pmin( cf.s.T1, cf.s.T2)
cf.s.E <- rep(0, n)
cf.s.E[cf.s.T1<=cf.s.T2] <- 1
cf.s.E[cf.s.T2<cf.s.T1] <- 2
exp.z <- exp(0.5 + 1*c1 - 1*c2)
pr <- exp.z/(1+exp.z)
TRT <- ifelse(runif(n)< pr, 1, 0)
```

```
X <- ifelse(TRT==1, cf.m.T, cf.s.T)
E <- ifelse(TRT==1, cf.m.E, cf.s.E)
covs.names <- c("c1", "c2")
data <- data.frame(X=X, E=E, TRT=TRT, c1=c1, c2=c2)
wei <- get.weights(data, "TRT", covs.names, wtype = "overlap")
hist(wei$ps[data$TRT==1], col="red", breaks = seq(0,1,0.05))
par(new=TRUE)
hist(wei$ps[data$TRT==0], col="blue", breaks = seq(0,1,0.05))
# Nonparametric estimation:
res.ATE <- fit.nonpar(df=data, X="X", E="E", A="TRT", C=covs.names, wtype="stab.ATE")
nonpar.pe <- get.pointEst(res.ATE, 0.5)
nonpar.pe$trt.eff[[1]]$RD # -0.1776143

# please see our package vignette for practical examples
```

`get.numAtRisk` *Number-at-risk in raw and weighted data*

Description

Obtaining time-varying number-at-risk statistic in both raw and weighted data

Usage

```
get.numAtRisk(df, X, E, A, C = NULL, wtype = "unadj", cens = 0)
```

Arguments

`df`	a data frame that includes time-to-event `X`, type of event `E`, a treatment indicator `A` and covariates `C`.
`X`	a character string specifying the name of the time-to-event variable in `df`.
`E`	a character string specifying the name of the "event type" variable in `df`.
`A`	a character specifying the name of the treatment/exposure variable. It is assumed that `A` is a numeric binary indicator with 0/1 values, where `A`=1 is assumed a treatment group, and `A`=0 a control group.
`C`	a vector of character strings with variable names (potential confounders) in the logistic regression model for Propensity Scores, i.e. P(A=1|C=c). The default value of `C` is NULL corresponding to `wtype`="unadj" that will estimate treatment effects in the raw (observed) data.
`wtype`	a character string variable indicating the type of weights that will define the target population for which the ATE will be estimated. The default is "unadj" - this will not adjust for possible treatment selection bias and will not use propensity scores weighting. It can be used, for example, in data from a randized controlled trial (RCT) where there is no need for emulation of baseline randomization. Other possible values are "stab.ATE", "ATE", "ATT", "ATC" and "overlap". See Table 1 from Li, Morgan, and Zaslavsky (2018). "stab.ATE" is defined as P(A=a)/P(A=a|C=c) - see Hernán et al. (2000).

cens — an integer value in E that corresponds to censoring times recorded in X. By default fit.nonpar assumes cens=0

Value

A list with two fields:

- trt.0 a matrix with three columns, time, num and sample corresponding to the treatment arm with A=0. The results for both weighted and unadjusted number-at-risk are returnd in a long-format matrix. The column time is a vector of time points at which we calculate the number-at-risk. The column num is the number-at-risk. The column sample is a factor variable that gets one of two values, "Weighted" or "Unadjusted". The estimated number-at-risk in the weighted sample corresponds to the rows with sample="Weighted".
- trt.1 a matrix with three columns, time, num and sample corresponding to the treatment arm with A=1. The results for both weighted and unadjusted number-at-risk are returnd in a long-format matrix. The column time is a vector of time points at which we calculate the number-at-risk. The column num is the number-at-risk. The column sample is a factor variable that gets one of two values, "Weighted" or "Unadjusted". The estimated number-at-risk in the weighted sample corresponds to the rows with sample="Weighted".

See Also

get.weights, get.pointEst, causalCmprsk

Examples

```
# create a data set
n <- 1000
set.seed(7)
c1 <- runif(n)
c2 <- as.numeric(runif(n)< 0.2)
set.seed(77)
cf.m.T1 <- rweibull(n, shape=1, scale=exp(-(-1 + 2*c1)))
cf.m.T2 <-  rweibull(n, shape=1, scale=exp(-(1 + 1*c2)))
cf.m.T <- pmin( cf.m.T1, cf.m.T2)
cf.m.E <- rep(0, n)
cf.m.E[cf.m.T1<=cf.m.T2] <- 1
cf.m.E[cf.m.T2<cf.m.T1] <- 2
set.seed(77)
cf.s.T1 <- rweibull(n, shape=1, scale=exp(-1*c1 ))
cf.s.T2 <-  rweibull(n, shape=1, scale=exp(-2*c2))
cf.s.T <- pmin( cf.s.T1, cf.s.T2)
cf.s.E <- rep(0, n)
cf.s.E[cf.s.T1<=cf.s.T2] <- 1
cf.s.E[cf.s.T2<cf.s.T1] <- 2
exp.z <- exp(0.5 + 1*c1 - 1*c2)
pr <- exp.z/(1+exp.z)
TRT <- ifelse(runif(n)< pr, 1, 0)
X <- ifelse(TRT==1, cf.m.T, cf.s.T)
E <- ifelse(TRT==1, cf.m.E, cf.s.E)
covs.names <- c("c1", "c2")
```

```
data <- data.frame(X=X, E=E, TRT=TRT, c1=c1, c2=c2)

num.atrisk <- get.numAtRisk(data, "X", "E", "TRT", C=covs.names, wtype="overlap", cens=0)
plot(num.atrisk$trt.1$time[num.atrisk$trt.1$sample=="Weighted"],
     num.atrisk$trt.1$num[num.atrisk$trt.1$sample=="Weighted"], col="red", type="s",
     xlab="time", ylab="number at risk",
     main="Number at risk in TRT=1", ylim=c(0, max(num.atrisk$trt.1$num)))
lines(num.atrisk$trt.1$time[num.atrisk$trt.1$sample=="Unadjusted"],
      num.atrisk$trt.1$num[num.atrisk$trt.1$sample=="Unadjusted"], col="blue", type="s")
legend("topright", legend=c("Weighted", "Unadjusted"), lty=1:1,  col=c("red", "blue"))
plot(num.atrisk$trt.0$time[num.atrisk$trt.0$sample=="Weighted"],
     num.atrisk$trt.0$num[num.atrisk$trt.0$sample=="Weighted"], col="red", type="s",
     xlab="time", ylab="number at risk",
     main="Number at risk in TRT=0", ylim=c(0, max(num.atrisk$trt.0$num)))
lines(num.atrisk$trt.0$time[num.atrisk$trt.0$sample=="Unadjusted"],
      num.atrisk$trt.0$num[num.atrisk$trt.0$sample=="Unadjusted"], col="blue", type="s")
legend("topright", legend=c("Weighted", "Unadjusted"), lty=1:1,  col=c("red", "blue"))
```

`get.pointEst` *Returns point estimates corresponding to a specific time point*

Description

Returns point estimates corresponding to a specific time point

Usage

```
get.pointEst(cmprsk.obj, timepoint)
```

Arguments

`cmprsk.obj`	a "cmprsk" object returned by one of the function `fit.cox` or `fit.nonpar`
`timepoint`	a scalar value of the time point of interest

Value

A list with the following fields:

`time`
a scalar timepoint passed into the function
`trt.0`
a list of estimates of the absolute counterfactual parameters corresponding to `A=0` and the type of event `E`. `trt.0` has the numb

- `CumHaz` point cumulative hazard estimates
- `CIF` point cumulative incidence function estimated
- `RMT` point estimates of the restricted mean time

`trt.1`
a list of estimates of the absolute counterfactual parameters corresponding to `A=1` and the type of event `E`. `trt.1` has the numb

- `CumHaz` point cumulative hazard estimates
- `CIF` point cumulative incidence function estimated
- `RMT` point estimates of the restricted mean time

`trt.eff`
a list of estimates of the treatment effect measures corresponding to the type of event `E`. `trt.eff` has the number of fields as th

- `log.CumHazR` a point estimate of the log of the ratio of hazards in two treatment arms
- `RD` a point estimate of the risk difference between two treatment arms
- `RR` a point estimate of the risk ratio between two treatment arms
- `ATE.RMT` a point estimate of the restricted mean time difference between two treatment arms

See Also

`fit.cox`, `fit.nonpar`, `causalCmprsk`

Examples

```
# create a data set
n <- 1000
set.seed(7)
c1 <-  runif(n)
c2 <- as.numeric(runif(n)< 0.2)
set.seed(77)
cf.m.T1 <- rweibull(n, shape=1, scale=exp(-(-1 + 2*c1)))
cf.m.T2 <-  rweibull(n, shape=1, scale=exp(-(1 + 1*c2)))
cf.m.T <- pmin( cf.m.T1, cf.m.T2)
cf.m.E <- rep(0, n)
cf.m.E[cf.m.T1<=cf.m.T2] <- 1
cf.m.E[cf.m.T2<cf.m.T1] <- 2
set.seed(77)
cf.s.T1 <- rweibull(n, shape=1, scale=exp(-1*c1 ))
cf.s.T2 <-  rweibull(n, shape=1, scale=exp(-2*c2))
cf.s.T <- pmin( cf.s.T1, cf.s.T2)
cf.s.E <- rep(0, n)
cf.s.E[cf.s.T1<=cf.s.T2] <- 1
cf.s.E[cf.s.T2<cf.s.T1] <- 2
exp.z <- exp(0.5 + 1*c1 - 1*c2)
pr <- exp.z/(1+exp.z)
TRT <- ifelse(runif(n)< pr, 1, 0)
X <- ifelse(TRT==1, cf.m.T, cf.s.T)
E <- ifelse(TRT==1, cf.m.E, cf.s.E)
covs.names <- c("c1", "c2")
data <- data.frame(X=X, E=E, TRT=TRT, c1=c1, c2=c2)
```

```
wei <- get.weights(data, "TRT", covs.names, wtype = "overlap")
hist(wei$ps[data$TRT==1], col="red", breaks = seq(0,1,0.05))
par(new=TRUE)
hist(wei$ps[data$TRT==0], col="blue", breaks = seq(0,1,0.05))
# Nonparametric estimation:
res.ATE <- fit.nonpar(df=data, X="X", E="E", A="TRT", C=covs.names, wtype="stab.ATE")
nonpar.pe <- get.pointEst(res.ATE, 0.5)
nonpar.pe$trt.eff[[1]]$RD
# Cox-based estimation:
res.cox.ATE <- fit.cox(df=data, X="X", E="E", A="TRT", C=covs.names, wtype="stab.ATE")
cox.pe <- get.pointEst(res.cox.ATE, 0.5)
cox.pe$trt.eff[[1]]$RD

# please see our package vignette for practical examples
```

`get.weights` *Fitting a logistic regression model for propensity scores and estimating weights*

Description

Fits a propensity scores model by logistic regression and returns both estimated propensity scores and requested weights. The estimated propensity scores can be used for further diagnostics, e.g. for testing a positivity assumption and covariate balance.

Usage

```
get.weights(df, A, C, wtype = "stab.ATE", case.w = NULL)
```

Arguments

`df`	a data frame that includes a treatment indicator `A` and covariates `C`.
`A`	a character specifying the name of the treatment/exposure variable. It is assumed that `A` is a numeric binary indicator with 0/1 values, where `A`=1 is assumed a treatment group, and `A`=0 a control group.
`C`	a vector of character strings with variable names (potential confounders) in the logistic regression model for Propensity Scores, i.e. P(A=1|C=c). The default value of `C` is NULL corresponding to `wtype`="unadj" that will estimate treatment effects in the raw (observed) data.
`wtype`	a character string variable indicating the type of weights that will define the target population for which the ATE will be estimated. The default is "stab.ATE" defined as P(A=a)/P(A=a|C=c) - see Hernán et al. (2000). Other possible values are "ATE", "ATT", "ATC", and "overlap". See Table 1 from Li, Morgan, and Zaslavsky (2018).
`case.w`	a vector of case weights.

Value

A list with the following fields:

- wtype a character string indicating the type of the estimated weights
- ps a vector of estimated propensity scores P(A=a|C=c)
- w a vector of estimated weights
- summary.glm a summary of the logistic regression fit which is done using stats::glm

function

References

F. Li, K.L. Morgan, and A.M. Zaslavsky. 2018. Balancing Covariates via Propensity Score Weighting. Journal of the American Statistical Association 113 (521): 390–400.

M.A. Hernán, B. Brumback, and J.M. Robins. 2000. Marginal structural models and to estimate the causal effect of zidovudine on the survival of HIV-positive men. Epidemiology, 11 (5): 561-570.

See Also

fit.nonpar, fit.cox, causalCmprsk

Examples

```
# create a data set
n <- 1000
set.seed(7)
c1 <- runif(n)
c2 <- as.numeric(runif(n)< 0.2)
set.seed(77)
cf.m.T1 <- rweibull(n, shape=1, scale=exp(-(-1 + 2*c1)))
cf.m.T2 <-  rweibull(n, shape=1, scale=exp(-(1 + 1*c2)))
cf.m.T <- pmin( cf.m.T1, cf.m.T2)
cf.m.E <- rep(0, n)
cf.m.E[cf.m.T1<=cf.m.T2] <- 1
cf.m.E[cf.m.T2<cf.m.T1] <- 2
set.seed(77)
cf.s.T1 <- rweibull(n, shape=1, scale=exp(-1*c1 ))
cf.s.T2 <-  rweibull(n, shape=1, scale=exp(-2*c2))
cf.s.T <- pmin( cf.s.T1, cf.s.T2)
cf.s.E <- rep(0, n)
cf.s.E[cf.s.T1<=cf.s.T2] <- 1
cf.s.E[cf.s.T2<cf.s.T1] <- 2
exp.z <- exp(0.5 + 1*c1 - 1*c2)
pr <- exp.z/(1+exp.z)
TRT <- ifelse(runif(n)< pr, 1, 0)
X <- ifelse(TRT==1, cf.m.T, cf.s.T)
E <- ifelse(TRT==1, cf.m.E, cf.s.E)
covs.names <- c("c1", "c2")
data <- data.frame(X=X, E=E, TRT=TRT, c1=c1, c2=c2)
wei <- get.weights(data, "TRT", covs.names, wtype = "overlap")
```

```
hist(wei$ps[data$TRT==1], col="red", breaks = seq(0,1,0.05))
par(new=TRUE)
hist(wei$ps[data$TRT==0], col="blue", breaks = seq(0,1,0.05))

# please see our package vignette for practical examples
```

Index