Package 'causal.decomp'

March 3, 2023

Title Causal Decomposition Analysis

Version 0.1.0

Date 2023-3-2

Depends R (>= 2.10)

Imports stats, parallel, MASS, nnet, SuppDists, CBPS, PSweight, spelling, utils

Suggests knitr, rmarkdown

VignetteBuilder knitr

Description We implement causal decomposition analysis using the methods proposed by Park, Lee, and Qin (2020) and Park, Kang, and Lee (2021+) <arXiv:2109.06940>. This package allows researchers to use the multiple-mediator-imputation, single-mediator-imputation, and product-of-coefficients regression methods to estimate the initial disparity, disparity reduction, and disparity remaining. It also allows to make the inference conditional on baseline covariates. We also implement sensitivity analysis for the causal decomposition analysis using R-squared values as sensitivity parameters (Park, Kang, Lee, and Ma, 2023).

License GPL-2

Encoding UTF-8

RoxygenNote 7.2.3

LazyData true

NeedsCompilation no

Author Suyeon Kang [aut, cre], Soojin Park [aut]

Maintainer Suyeon Kang <skang062@ucr.edu>

Repository CRAN

Date/Publication 2023-03-03 22:50:12 UTC

R topics documented:

`causal.decomp` *causal.decomp: Causal Decomposition Analysis.*

Description

The causal.decomp package provides four important functions: mmi, smi, pocr, and sensitivity.

`mmi` *Multiple-Mediator-Imputation Estimation Method*

Description

'mmi' is used to estimate the initial disparity, disparity reduction, and disparity remaining for causal decomposition analysis, using the multiple-mediator-imputation estimation method proposed by Park et al. (2020).

Usage

```
mmi(fit.r = NULL, fit.x, fit.y, treat, covariates, sims = 100, conf.level = .95,
    conditional = TRUE, cluster = NULL, long = TRUE, mc.cores = 1L, seed = NULL)
```

Arguments

`fit.r`	a fitted model object for treatment. Can be of class 'CBPS' or 'SumStat'. Default is 'NULL'. Only necessary if 'conditional' is 'FALSE'.
`fit.x`	a fitted model object for intermediate confounder(s). Each intermediate model can be of class 'lm', 'glm', 'multinom', or 'polr'. When multiple confounders are considered, can be of class 'list' containing multiple models.
`fit.y`	a fitted model object for outcome. Can be of class 'lm' or 'glm'.
`treat`	a character string indicating the name of the treatment variable used in the models. The treatment can be categorical with two or more categories (two- or multi-valued factor).
`covariates`	a vector containing the name of the covariate variable(s) used in the models. Each covariate can be categorical with two or more categories (two- or multi-valued factor) or continuous (numeric).
`sims`	number of Monte Carlo draws for nonparametric bootstrap.

conf.level	level of the returned two-sided confidence intervals, which are estimated by the nonparametric percentile bootstrap method. Default is .95, which returns the 2.5 and 97.5 percentiles of the simulated quantities.
conditional	a logical value. If 'TRUE', the function will return the estimates conditional on those covariate values, and all covariates in mediator and outcome models need to be centered prior to fitting. Default is 'TRUE'. If 'FALSE', 'fit.r' needs to be specified.
cluster	a vector of cluster indicators for the bootstrap. If provided, the cluster bootstrap is used. Default is 'NULL'.
long	a logical value. If 'TRUE', the output will contain the entire sets of estimates for all bootstrap samples. Default is 'TRUE'.
mc.cores	The number of cores to use. Must be exactly 1 on Windows.
seed	seed number for the reproducibility of results. Default is 'NULL'.

Details

This function returns the point estimates of the initial disparity, disparity reduction, and disparity remaining for a categorical treatment and a variety of types of outcome and mediator(s) in causal decomposition analysis. It also returns nonparametric percentile bootstrap confidence intervals for each estimate.

The initial disparity represents the expected difference in an outcome between a comparison group $R = j$ and a reference group $R = i$ where $i \neq j$. That is,

$$\tau(i, j) \ = \ E\{Y|R = j\} - E\{Y|R = i\},$$

where R and Y are the group indicator and the outcome variable, respectively. The disparity reduction represents the expected change in an outcome for the group $R = j$ after adjusting the level of mediator(s) to the level of the reference group. That is,

$$\delta(j) \ = \ E\{Y|R = j\} - E\{Y(G_M(i))|R = j\},$$

where $G_M(i)$ is a random draw from the mediator distribution of the reference group. The disparity remaining represents the remaining disparity for the group $R = j$ even after adjusting the level of mediators to the reference group. Formally,

$$\zeta(i) \ = \ E\{Y(G_M(i))|R = j\} - E\{Y|R = i\}.$$

The disparity reduction and remaining can be estimated using the multiple-mediator-imputation method suggested by Park et al. (2020). See the references for more details.

If one wants to make the inference conditional on baseline covariates, set 'conditional = TRUE' and center the data before fitting the models.

As of version 0.1.0, the intetmediate confounder model ('fit.x') can be of class 'lm', 'glm', 'multinom', or 'polr', corresponding respectively to the linear regression models and generalized linear models, multinomial log-linear models, and ordered response models. The outcome model ('fit.y') can be of class 'lm' or 'glm'. Also, the treatment model ('fit.r') can be of class 'CBPS' or 'SumStat', both of which use the propensity score weighting. It is only necessary when 'conditional = FALSE'.

Value

result	a matrix containing the point estimates of the initial disparity, disparity remaining, and disparity reduction, and the percentile bootstrap confidence intervals for each estimate.
all.result	a matrix containing the point estimates of the initial disparity, disparity remaining, and disparity reduction for all bootstrap samples. Returned if 'long' is 'TRUE'.

Author(s)

Suyeon Kang, University of California, Riverside, <skang062@ucr.edu>; Soojin Park, University of California, Riverside, <soojinp@ucr.edu>.

References

Park, S., Lee, C., and Qin, X. (2020). "Estimation and sensitivity analysis for causal decomposition in heath disparity research", Sociological Methods & Research, 00491241211067516.

Park, S., Kang, S., and Lee, C. (2021+). "Choosing an optimal method for causal decomposition analysis: A better practice for identifying contributing factors to health disparities". arXiv preprint arXiv:2109.06940.

See Also

smi

Examples

```
data(sdata)

#-------------------------------------------------------------------------------#
# Example 1-a: Continuous Outcome
#-------------------------------------------------------------------------------#
fit.m1 <- lm(M.num ~ R + C.num + C.bin, data = sdata)
fit.m2 <- glm(M.bin ~ R + C.num + C.bin, data = sdata,
          family = binomial(link = "logit"))
require(MASS)
fit.m3 <- polr(M.cat ~ R + C.num + C.bin, data = sdata)
fit.x1 <- lm(X ~ R + C.num + C.bin, data = sdata)
require(nnet)
fit.m4 <- multinom(M.cat ~ R + C.num + C.bin, data = sdata)
fit.y1 <- lm(Y.num ~ R + M.num + M.bin + M.cat + X + C.num + C.bin,
          data = sdata)

require(PSweight)
fit.r1 <- SumStat(R ~ C.num + C.bin, data = sdata, weight = "IPW")
require(CBPS)
fit.r2 <- CBPS(R ~ C.num + C.bin, data = sdata, method = "exact",
          standardize = "TRUE")

res.1a <- mmi(fit.r = fit.r1, fit.x = fit.x1,
```

```
          fit.y = fit.y1, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.1a

#-------------------------------------------------------------------------------#
# Example 1-b: Binary Outcome
#-------------------------------------------------------------------------------#
fit.y2 <- glm(Y.bin ~ R + M.num + M.bin + M.cat + X + C.num + C.bin,
          data = sdata, family = binomial(link = "logit"))

res.1b <- mmi(fit.r = fit.r1, fit.x = fit.x1,
          fit.y = fit.y2, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.1b

#-------------------------------------------------------------------------------#
# Example 2-a: Continuous Outcome, Conditional on Covariates
#-------------------------------------------------------------------------------#
# For conditional = TRUE, need to create data with centered covariates
# copy data
sdata.c <- sdata
# center quantitative covariate(s)
sdata.c$C.num <- scale(sdata.c$C.num, center = TRUE, scale = FALSE)
# center binary (or categorical) covariates(s)
# only neccessary if the desired baseline level is NOT the default baseline level.
sdata.c$C.bin <- relevel(sdata.c$C.bin, ref = "1")

# fit mediator and outcome models
fit.m1 <- lm(M.num ~ R + C.num + C.bin, data = sdata.c)
fit.m2 <- glm(M.bin ~ R + C.num + C.bin, data = sdata.c,
          family = binomial(link = "logit"))
fit.m3 <- polr(M.cat ~ R + C.num + C.bin, data = sdata.c)
fit.x2 <- lm(X ~ R + C.num + C.bin, data = sdata.c)
fit.y1 <- lm(Y.num ~ R + M.num + M.bin + M.cat + X + C.num + C.bin,
          data = sdata.c)

res.2a <- mmi(fit.x = fit.x2,
          fit.y = fit.y1, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2a

#-------------------------------------------------------------------------------#
# Example 2-b: Binary Outcome, Conditional on Covariates
#-------------------------------------------------------------------------------#
fit.y2 <- glm(Y.bin ~ R + M.num + M.bin + M.cat + X + C.num + C.bin,
          data = sdata.c, family = binomial(link = "logit"))

res.2b <- mmi(fit.x = fit.x2,
          fit.y = fit.y2, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2b
```

plot *Visualize sensitivity Objects*

Description

S3 methods visualizing results for some objects generated by `sensitivity`.

Usage

```
## S3 method for class 'sensitivity'
plot(x, xlim = c(0, 0.3), ylim = c(0, 0.3), ...)
```

Arguments

`x`	an object of class `sensitivity`.
`xlim`	limits of the x-axis of contour plots.
`ylim`	limits of the y-axis of contour plots.
`...`	Other arguments for future usage.

pocr *Product-of-Coefficients-Regression Estimation Method*

Description

'pocr' is used to estimate the initial disparity, disparity reduction, and disparity remaining for causal decomposition analysis, using the product-of-coefficients-regression estimation method proposed by Park et al. (2021+).

Usage

```
pocr(fit.x = NULL, fit.m, fit.y, treat, covariates, sims = 100, conf.level = .95,
     cluster = NULL, long = TRUE, mc.cores = 1L, seed = NULL)
```

Arguments

`fit.x`	a fitted model object for intermediate confounder. Can be of class 'lm'. Only necessary if the mediator is categorical. Default is 'NULL'.
`fit.m`	a fitted model object for mediator. Can be of class 'lm', 'glm', 'multinom', or 'polr'.
`fit.y`	a fitted model object for outcome. Can be of class 'lm'.
`treat`	a character string indicating the name of the treatment variable used in the models. The treatment can be categorical with two or more categories (two- or multi-valued factor).

covariates	a vector containing the name of the covariate variable(s) used in the models. Each covariate can be categorical with two or more categories (two- or multi-valued factor) or continuous (numeric).
sims	number of Monte Carlo draws for nonparametric bootstrap.
conf.level	level of the returned two-sided confidence intervals, which are estimated by the nonparametric percentile bootstrap method. Default is to return the 2.5 and 97.5 percentiles of the simulated quantities.
cluster	a vector of cluster indicators for the bootstrap. If provided, the cluster bootstrap is used. Default is 'NULL'.
long	a logical value. If 'TRUE', the output will contain the entire sets of estimates for all bootstrap samples. Default is 'TRUE'.
mc.cores	The number of cores to use. Must be exactly 1 on Windows.
seed	seed number for the reproducibility of results. Default is 'NULL'.

Details

This function returns the point estimates of the initial disparity, disparity reduction, and disparity remaining for a categorical treatment and a variety of types of outcome and mediator(s) in causal decomposition analysis. It also returns nonparametric percentile bootstrap confidence intervals for each estimate.

The definition of the initial disparity, disparity reduction, and disparity remaining can be found in help('mmi'). As opposed to the 'mmi' and 'smi' function, this function uses the product-of-coefficients-regression method suggested by Park et al. (2021+). It always make the inference conditional on baseline covariates. Therefore, users need to center the data before fitting the models. See the reference for more details.

As of version 0.1.0, the mediator model ('fit.m') can be of class 'lm', 'glm', 'multinom', or 'polr', corresponding respectively to the linear regression models and generalized linear models, multinomial log-linear models, and ordered response models. The outcome model ('fit.y') can be of class 'lm'. The intermediate confounder model ('fit.x') can also be of class 'lm' and only necessary when the mediator is categorical.

Value

result	a matrix containing the point estimates of the initial disparity, disparity remaining, and disparity reduction, and the percentile bootstrap confidence intervals for each estimate.
all.result	a matrix containing the point estimates of the initial disparity, disparity remaining, and disparity reduction for all bootstrap samples. Returned if 'long' is 'TRUE'.

Author(s)

Suyeon Kang, University of California, Riverside, <skang062@ucr.edu>; Soojin Park, University of California, Riverside, <soojinp@ucr.edu>.

References

Park, S., Kang, S., and Lee, C. (2021+). "Choosing an optimal method for causal decomposition analysis: A better practice for identifying contributing factors to health disparities". arXiv preprint arXiv:2109.06940.

See Also

mmi, smi

Examples

```
data(sdata)

# To be conditional on covariates, first create data with centered covariates
# copy data
sdata.c <- sdata
# center quantitative covariate(s)
sdata.c$C.num <- scale(sdata.c$C.num, center = TRUE, scale = FALSE)
# center binary (or categorical) covariates(s)
# only neccessary if the desired baseline level is NOT the default baseline level.
sdata.c$C.bin <- relevel(sdata.c$C.bin, ref = "1")

#-------------------------------------------------------------------------------#
# Example 1: Continuous Mediator
#-------------------------------------------------------------------------------#
fit.m1 <- lm(M.num ~ R + C.num + C.bin, data = sdata.c)
fit.y1 <- lm(Y.num ~ R + M.num + M.num:R + X +
          C.num + C.bin, data = sdata.c)
res1 <- pocr(fit.m = fit.m1, fit.y = fit.y1, sims = 40,
        covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res1

#-------------------------------------------------------------------------------#
# Example 2: Binary Mediator
#-------------------------------------------------------------------------------#
fit.x1 <- lm(X ~ R + C.num + C.bin, data = sdata.c)
fit.m2 <- glm(M.bin ~ R + C.num + C.bin, data = sdata.c,
          family = binomial(link = "logit"))
fit.y2 <- lm(Y.num ~ R + M.bin + M.bin:R + X +
          C.num + C.bin, data = sdata.c)
res2 <- pocr(fit.x = fit.x1, fit.m = fit.m2, fit.y = fit.y2,
        sims = 40, covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res2

#-------------------------------------------------------------------------------#
# Example 3: Ordinal Mediator
#-------------------------------------------------------------------------------#
require(MASS)
fit.m3 <- polr(M.cat ~ R + C.num + C.bin, data = sdata.c)
fit.y3 <- lm(Y.num ~ R + M.cat + M.cat:R + X +
C.num + C.bin, data = sdata.c)
res3 <- pocr(fit.x = fit.x1, fit.m = fit.m3, fit.y = fit.y3,
```

```
       sims = 40, covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res3

#-------------------------------------------------------------------------------#
# Example 4: Nominal Mediator
#-------------------------------------------------------------------------------#
require(nnet)
fit.m4 <- multinom(M.cat ~ R + C.num + C.bin, data = sdata.c)
res4 <- pocr(fit.x = fit.x1, fit.m = fit.m4, fit.y = fit.y3,
        sims = 40, covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res4
```

`sdata` *Synthetic Data for Illustration*

Description

A randomly generated dataset containing 1000 rows and 9 columns with no missing values.

Usage

```
sdata
```

Format

A data frame containing the following variables. The data are provided only for explanatory purposes. The mediators are assumed to be independent of each other.

C.num: A quantitative covariate.

C.bin: A binary covariates with a value of 0 or 1.

R: A group indicator with four levels.

X: A quantitative intermediate confounder between a mediator and the outcome.

M.num: A quantitative mediator.

M.bin: A binary mediator with a value of 0 or 1.

M.cat: A categorical mediator with three levels.

Y.num: A quantitative outcome.

Y.bin: A binary outcome with a value of 0 or 1.

Details

Note that all the variables are randomly generated using the dataset used in Park et al. (2021+).

References

Park, S., Kang, S., and Lee, C. (2021+). "Choosing an Optimal Method for Causal Decomposition Analysis: A Better Practice for Identifying Contributing Factors to Health Disparities".

`sensitivity` *Sensitivity Analysis Using R-Squared Values for Causal Decomposition Analysis*

Description

The function 'sensitivity' is used to implement sensitivity analysis for the causal decomposition analysis, using R-squared values as sensitivity parameters.

Usage

```
sensitivity(boot.res, fit.y, fit.m = NULL, mediator = NULL, covariates, treat,
            sel.lev.treat, conf.level = 0.95, max.rsq = 0.3)
```

Arguments

`boot.res`	bootstrap results from an object fitted by the 'smi' function.
`fit.y`	outcome model used in fitting the 'smi' function. Can be of class 'lm' or 'glm'.
`fit.m`	mediator model used in fitting the 'smi' function. Can be of class 'lm', 'glm', or 'polr'.
`mediator`	a vector containing the name of mediator used in the models.
`covariates`	a vector containing the name of the covariate variable(s) used in the models. Each covariate can be categorical with two or more categories (two- or multi-valued factor) or continuous (numeric).
`treat`	a character string indicating the name of the treatment variable used in the models. The treatment can be categorical with two or more categories (two- or multi-valued factor).
`sel.lev.treat`	a level of categorical treatment variable which is to be compared with the reference level.
`conf.level`	level of the returned two-sided confidence intervals, which are estimated by the nonparametric percentile bootstrap method. Default is .95, which returns the 2.5 and 97.5 percentiles of the simulated quantities.
`max.rsq`	upper limit of the two sensitivity parameters (R-squared values). Once it is set, the R-squared values between 0 and this upper limit are explored to draw the sensitivity contour plots. Default is 0.3.

Details

This function is used to implement sensitivity analysis for the causal decomposition analysis, using two sensitivity parameters: (i) the R-squared value between the outcome and unobserved confounder given the treatment, intermediate confounder, mediator, and covariates; and (ii) the R-squared value between the mediator and unobserved confounder given the treatment, intermediate confounder, and covariates (Park et al., 2023).

As of version 0.1.0, 'boot.res' must be fitted by the 'smi' function with a single mediator, which can be of class 'lm', 'glm', or 'polr'.

Value

`call` original function call.

`disparity.reduction`
a matrix containing the estimated disparity reduction value along with lower and upper limits of the confidence interval, for each combination of the two sensitivity parameters, assuming those two are equal.

`disparity.remaining`
a matrix containing the estimated disparity remaining value along with lower and upper limits of the confidence interval, for each combination of the two sensitivity parameters, assuming those two are equal.

`rm` R-squared values between the mediator and unobserved confounder (first sensitivity parameter), which are explored for the sensitivity analysis.

`ry` R-squared values between the outcome and unobserved confounder, (second sensitivity parameter), which are explored for the sensitivity analysis.

`red` a matrix containing the estimated disparity reduction values given each combination of the two sensitivity parameters.

`lower_red` a matrix containing the lower limit of disparity reduction given each combination of the two sensitivity parameters.

`rem` a matrix containing the estimated disparity remaining values given each combination of the two sensitivity parameters.

`lower_rem` a matrix containing the lower limit of disparity remaining given each combination of the two sensitivity parameters.

`RV_red` robustness value for disparity reduction, which represents the strength of association that will explain away the estimated disparity reduction.

`RV_red_alpha` robustness value for disparity reduction, which represents the strength of association that will change the significance of the estimated disparity reduction at the given significance level, assuming an equal association to the mediator and the outcome.

`RV_rem` robustness value for disparity remaining, which represents the strength of association that will explain away the estimated disparity remaining.

`RV_rem_alpha` robustness value for disparity remaining, which represents the strength of association that will change the significance of the estimated disparity remaining at the given significance level, assuming an equal association to the mediator and the outcome.

Author(s)

Suyeon Kang, University of California, Riverside, <skang062@ucr.edu>; Soojin Park, University of California, Riverside, <soojinp@ucr.edu>.

References

Park, S., Kang, S., Lee, C., & Ma, S. (2023). Sensitivity analysis for causal decomposition analysis: Assessing robustness toward omitted variable bias, Journal of Causal Inference. Forthcoming.

See Also

smi

Examples

```
data(sMIDUS)

# Center covariates
sMIDUS$age <- scale(sMIDUS$age, center = TRUE, scale = FALSE)
sMIDUS$stroke <- scale(sMIDUS$stroke, center = TRUE, scale = FALSE)
sMIDUS$T2DM <- scale(sMIDUS$T2DM, center = TRUE, scale = FALSE)
sMIDUS$heart <- scale(sMIDUS$heart, center = TRUE, scale = FALSE)

fit.m <- lm(edu ~ racesex + age + stroke + T2DM + heart, data = sMIDUS)
fit.y <- lm(health ~ racesex + lowchildSES + abuse + edu + racesex:edu +
            age + stroke + T2DM + heart, data = sMIDUS)
smiRes <- smi(fit.m = fit.m, fit.y = fit.y, sims = 40, conf.level = .95,
         covariates = c("age", "stroke", "T2DM", "heart"), treat = "racesex", seed = 227)
sensRes <- sensitivity(boot.res = smiRes, fit.m = fit.m, fit.y = fit.y, mediator = "edu",
                     covariates = c("age", "stroke", "T2DM", "heart"), treat = "racesex",
                       sel.lev.treat = "4", max.rsq = 0.3)

sensRes
names(sensRes)
# Create sensitivity contour plots
plot(sensRes)
```

---

smi *Single-Mediator-Imputation Estimation Method*

Description

'smi' is used to estimate the initial disparity, disparity reduction, and disparity remaining for causal decomposition analysis, using the single-mediator-imputation estimation method proposed by Park et al. (2021+).

Usage

```
smi(fit.r = NULL, fit.m, fit.y, treat, covariates, sims = 100, conf.level = .95,
    conditional = TRUE, cluster = NULL, long = TRUE, mc.cores = 1L, seed = NULL)
```

Arguments

fit.r	a fitted model object for treatment. Can be of class 'CBPS' or 'SumStat'. Default is 'NULL'. Only necessary if 'conditional' is 'FALSE'.
fit.m	a fitted model object for mediator. Can be of class 'lm', 'glm', 'multinom', or 'polr'.
fit.y	a fitted model object for outcome. Can be of class 'lm' or 'glm'.

treat	a character string indicating the name of the treatment variable used in the models. The treatment can be categorical with two or more categories (two- or multi-valued factor).
covariates	a vector containing the name of the covariate variable(s) used in the models. Each covariate can be categorical with two or more categories (two- or multi-valued factor) or continuous (numeric).
sims	number of Monte Carlo draws for nonparametric bootstrap.
conf.level	level of the returned two-sided confidence intervals, which are estimated by the nonparametric percentile bootstrap method. Default is .95, which returns the 2.5 and 97.5 percentiles of the simulated quantities.
conditional	a logical value. If 'TRUE', the function will return the estimates conditional on those covariate values; and all covariates in mediator and outcome models need to be centered prior to fitting. Default is 'TRUE'. If 'FALSE', 'fit.r' needs to be specified.
cluster	a vector of cluster indicators for the bootstrap. If provided, the cluster bootstrap is used. Default is 'NULL'.
long	a logical value. If 'TRUE', the output will contain the entire sets of estimates for all bootstrap samples. Default is 'TRUE'.
mc.cores	The number of cores to use. Must be exactly 1 on Windows.
seed	seed number for the reproducibility of results. Default is 'NULL'.

Details

This function returns the point estimates of the initial disparity, disparity reduction, and disparity remaining for a categorical treatment and a variety of types of outcome and mediator(s) in causal decomposition analysis. It also returns nonparametric percentile bootstrap confidence intervals for each estimate.

The definition of the initial disparity, disparity reduction, and disparity remaining can be found in help('mmi'). As opposed to the 'mmi' function, this function uses the single-mediator-imputation method suggested by Park et al. (2021+). See the reference for more details.

If one wants to make the inference conditional on baseline covariates, set 'conditional = TRUE' and center the data before fitting the models.

As of version 0.1.0, the mediator model ('fit.m') can be of class 'lm', 'glm', 'multinom', or 'polr', corresponding respectively to the linear regression models and generalized linear models, multinomial log-linear models, and ordered response models. The outcome model ('fit.y') can be of class 'lm' or 'glm'. Also, the treatment model ('fit.r') can be of class 'CBPS' or 'SumStat', both of which use the propensity score weighting. It is only necessary when 'conditional = FALSE'.

Value

result	a matrix containing the point estimates of the initial disparity, disparity remaining, and disparity reduction, and the percentile bootstrap confidence intervals for each estimate.
all.result	a matrix containing the point estimates of the initial disparity, disparity remaining, and disparity reduction for all bootstrap samples. Returned if 'long' is 'TRUE'.

alpha.r	a vector containing the estimates of the regression coefficient of the treatment in the mediator. Not needed unless sensitivity analysis is conducted afterwards.
se.gamma	a vector containing the estimates of standard error of the egression coefficient of the mediator in the outcome model. Not needed unless sensitivity analysis is conducted afterwards.

Author(s)

Suyeon Kang, University of California, Riverside, <skang062@ucr.edu>; Soojin Park, University of California, Riverside, <soojinp@ucr.edu>.

References

Park, S., Kang, S., and Lee, C. (2021+). "Choosing an optimal method for causal decomposition analysis: A better practice for identifying contributing factors to health disparities". arXiv preprint arXiv:2109.06940.

See Also

mmi, sensitivity

Examples

```
data(sdata)

#-------------------------------------------------------------------------------#
# Example 1-a: Continuous Outcome
#-------------------------------------------------------------------------------#
require(PSweight)
fit.r1 <- SumStat(R ~ C.num + C.bin, data = sdata, weight = "IPW")
require(CBPS)
fit.r2 <- CBPS(R ~ C.num + C.bin, data = sdata, method = "exact",
          standardize = "TRUE")

# Continuous mediator
fit.m1 <- lm(M.num ~ R + C.num + C.bin, data = sdata)
fit.y1 <- lm(Y.num ~ R + M.num + X + C.num + C.bin, data = sdata)
res.1a1 <- smi(fit.r = fit.r1, fit.m = fit.m1,
          fit.y = fit.y1, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 32)
res.1a1

# Binary mediator
fit.m2 <- glm(M.bin ~ R + C.num + C.bin, data = sdata,
          family = binomial(link = "logit"))
fit.y2 <- lm(Y.num ~ R + M.bin + X + C.num + C.bin, data = sdata)
res.1a2 <- smi(fit.r = fit.r1, fit.m = fit.m2,
          fit.y = fit.y2, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.1a2
```

```
# Categorical mediator
require(MASS)
fit.m3 <- polr(M.cat ~ R + C.num + C.bin, data = sdata)
fit.y3 <- lm(Y.num ~ R + M.cat + X + C.num + C.bin, data = sdata)
res.1a3 <- smi(fit.r = fit.r1, fit.m = fit.m3,
          fit.y = fit.y3, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.1a3

require(nnet)
fit.m4 <- multinom(M.cat ~ R + C.num + C.bin, data = sdata)
res.1a4 <- smi(fit.r = fit.r1, fit.m = fit.m4,
          fit.y = fit.y3, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.1a4

#------------------------------------------------------------------------------#
# Example 1-b: Binary Outcome
#------------------------------------------------------------------------------#
# Continuous mediator
fit.y1 <- glm(Y.bin ~ R + M.num + X + C.num + C.bin,
          data = sdata, family = binomial(link = "logit"))
res.1b1 <- smi(fit.r = fit.r1, fit.m = fit.m1,
          fit.y = fit.y1, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 32)
res.1b1

# Binary mediator
fit.y2 <- glm(Y.bin ~ R + M.bin + X + C.num + C.bin,
          data = sdata, family = binomial(link = "logit"))
res.1b2 <- smi(fit.r = fit.r1, fit.m = fit.m2,
          fit.y = fit.y2, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.1b2

# Categorical mediator
fit.y3 <- glm(Y.bin ~ R + M.cat + X + C.num + C.bin,
          data = sdata, family = binomial(link = "logit"))
res.1b3 <- smi(fit.r = fit.r1, fit.m = fit.m3,
          fit.y = fit.y3, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.1b3

res.1b4 <- smi(fit.r = fit.r1, fit.m = fit.m4,
          fit.y = fit.y3, sims = 40, conditional = FALSE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.1b4

#------------------------------------------------------------------------------#
# Example 2-a: Continuous Outcome, Conditional on Covariates
#------------------------------------------------------------------------------#
# For conditional = T, need to create data with centered covariates
# copy data
```

```
sdata.c <- sdata
# center quantitative covariate(s)
sdata.c$C.num <- scale(sdata.c$C.num, center = TRUE, scale = FALSE)
# center binary (or categorical) covariates(s)
# only neccessary if the desired baseline level is NOT the default baseline level.
sdata.c$C.bin <- relevel(sdata.c$C.bin, ref = "1")

# Continuous mediator
fit.y1 <- lm(Y.num ~ R + M.num + X + C.num + C.bin, data = sdata.c)
fit.m1 <- lm(M.num ~ R + C.num + C.bin, data = sdata.c)
res.2a1 <- smi(fit.m = fit.m1,
          fit.y = fit.y1, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2a1

# Binary mediator
fit.y2 <- lm(Y.num ~ R + M.bin + X + C.num + C.bin, data = sdata.c)
fit.m2 <- glm(M.bin ~ R + C.num + C.bin, data = sdata.c,
          family = binomial(link = "logit"))
res.2a2 <- smi(fit.m = fit.m2,
          fit.y = fit.y2, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2a2

# Categorical mediator
fit.y3 <- lm(Y.num ~ R + M.cat + X + C.num + C.bin, data = sdata.c)
fit.m3 <- polr(M.cat ~ R + C.num + C.bin, data = sdata.c)
res.2a3 <- smi(fit.m = fit.m3,
          fit.y = fit.y3, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2a3

fit.m4 <- multinom(M.cat ~ R + C.num + C.bin, data = sdata.c)
res.2a4 <- smi(fit.m = fit.m4,
          fit.y = fit.y3, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2a4

#-----------------------------------------------------------------------------#
# Example 2-b: Binary Outcome, Conditional on Covariates
#-----------------------------------------------------------------------------#
# Continuous mediator
fit.y1 <- glm(Y.bin ~ R + M.num + X + C.num + C.bin,
          data = sdata.c, family = binomial(link = "logit"))
res.2b1 <- smi(fit.m = fit.m1,
          fit.y = fit.y1, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2b1

# Binary mediator
fit.y2 <- glm(Y.bin ~ R + M.bin + X + C.num + C.bin,
          data = sdata.c, family = binomial(link = "logit"))
res.2b2 <- smi(fit.m = fit.m2,
```

```
          fit.y = fit.y2, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2b2

# Categorical mediator
fit.y3 <- glm(Y.bin ~ R + M.cat + X + C.num + C.bin,
          data = sdata.c, family = binomial(link = "logit"))
res.2b3 <- smi(fit.m = fit.m3,
          fit.y = fit.y3, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2b3

res.2b4 <- smi(fit.m = fit.m4,
          fit.y = fit.y3, sims = 40, conditional = TRUE,
          covariates = c("C.num", "C.bin"), treat = "R", seed = 111)
res.2b4
```

---

sMIDUS *Synthetic Data Generated Based on the Midlife Development in the U.S. (MIDUS) Study*

Description

This is a synthetic dataset that includes variables from the Midlife Development in the U.S. (MIDUS) study. It has been artificially generated based on the actual MIDUS data, which is not publicly available due to confidentiality concerns. The synthetic data set consists of 1948 rows and 9 columns, with no missing values.

Usage

```
sMIDUS
```

Format

A data frame containing the following variables.

health: cardiovascular health score.

racesex: race-gender groups with four levels (1: White men, 2: White women, 3: Black men, 4: Black women).

lowchildSES: socioeconomic status (SES) in the childhood.

abuse: adverse experience in the childhood.

edu: education level.

age: age.

stroke: genetic vulnerability with a value of 0 or 1.

T2DM: genetic vulnerability with a value of 0 or 1.

heart: genetic vulnerability with a value of 0 or 1.

Details

Note that all the variables are fabricated using the actual MIDUS data used in Park et al. (2023).

References

Park, S., Kang, S., Lee, C., & Ma, S. (2023). Sensitivity Analysis for Causal Decomposition Analysis: Assessing Robustness Toward Omitted Variable Bias, Journal of Causal Inference. Forthcoming.

Index