# Package 'CIMTx'

October 20, 2021

**Type** Package

**Title** Causal Inference for Multiple Treatments with a Binary Outcome

**Version** 1.0.0

**Description**
Different methods to conduct causal inference for multiple treatments with a binary outcome, including regression adjustment, vector matching, Bayesian additive regression trees, targeted maximum likelihood and inverse probability of treatment weighting using different generalized propensity score models such as multinomial logistic regression, generalized boosted models and super learner. For more details, see the paper by Hu et al. <doi:10.1177/0962280220921909>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.1.1

**Imports** nnet, BART, twang, arm, dplyr, Matching, magrittr, WeightIt, tmle, tidyr, stats, ggplot2, cowplot, mgcv, metR, stringr, SuperLearner, foreach, doParallel

**NeedsCompilation** no

**Author** Liangyuan Hu [aut],
Chenyang Gu [aut],
Michael Lopez [aut],
Jiayi Ji [aut, cre]

**Maintainer** Jiayi Ji <Jiayi.Ji@mountsinai.org>

**Repository** CRAN

**Date/Publication** 2021-10-20 07:10:05 UTC

## R topics documented:

---

`ce_estimate` *Causal inference with multiple treatments using observational data*

---

## Description

This function implements the 6 different methods for causal inference with multiple treatments using observational data.

## Usage

```
ce_estimate(
  y,
  x,
  w,
  method,
  discard = "No",
  estimand,
  trim_perc,
  SL.library,
  reference_trt,
  boot = FALSE,
  nboots,
  ndpost = 1000,
  caliper = 0.25,
  n_cluster = 5,
  ...
)
```

## Arguments

| | |
|---|---|
| y | a numeric vector (0, 1) representing a binary outcome |
| x | a dataframe, including all the covariates but not treatments. |
| w | a numeric vector representing the treatment groups |
| method | a character string. Users can selected from the following methods including "RA", "VM", "BART", "TMLE", "IPTW-Multinomial", "IPTW-GBM", "IPTW-SL", "RAMS-Multinomial", "RAMS-GBM", "RAMS-SL" |
| discard | "No" or "Yes" indicating whether to use the discarding rules for the BART based method. The default is "No" |
| estimand | "ATT" or "ATE" representing the type of causal estimand. When the estimand = "ATT", users also need to specify the reference treatment group by setting the reference_trt argument. |
| trim_perc | the percentile at which the inverse probability of treatment weights shouldbe trimmed |
| SL.library | a character vector of prediction algorithms. A list of functions included in the SuperLearner package can be found with listWrappers(). |
| reference_trt | reference treatment group for ATT effect |
| boot | is logical, indicating whether or not to use nonparametric bootstrap to calculate the 95% confidence intervals of the causal effect estimates. |
| nboots | only need to set up when the boot = TRUE; is a numeric value representing the number of bootstrap samples |
| ndpost | is the number of posterior draws for the Bayesian methods |
| caliper | only need to set up when the method is set to VM; is a numeric value denoting the caliper which should be used when matching on the logit of GPS within each cluster formed by K-means clustering. The caliper is in standardized units. For example, caliper = 0.25 means that all matches greater than 0.25 standard deviations of the logit of GPS are dropped. The default value is 0.25 |
| n_cluster | only need to set up when the method is set to VM; a numeric value denoting the number of clusters to form using K means clustering on the logit of GPS. The default value is 5. |
| ... | Other parameters that can be passed through the functions |

## Value

a list with w-1 elements for ATT effect; a list with w*(w-1)/2 elements for ATE effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
```

```
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate(y = data$y, x = data$covariates, w = data$w,
ndpost=100, method = "RA", estimand = "ATE")
```

---

`ce_estimate_bart_ate` *Bayesian Additive Regression Trees (BART) for ATE estimation*

---

## Description

This function implements the BART method when estimand is ATE. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_bart_ate(y, x, w, discard = "No", ndpost = 1000, ...)
```

## Arguments

| | |
|---|---|
| `y` | a numeric vector (0, 1) representing a binary outcome |
| `x` | a dataframe, including all the covariates but not treatments |
| `w` | a numeric vector representing the treatment groups |
| `discard` | "No" or "Yes" indicating whether to use the discarding rules for the BART based method. The default is "No" |
| `ndpost` | number of posterior samples from BART |
| `...` | Other parameters that can be passed through the BART::pbart() function |

## Value

a list with w*(w-1)/2 elements for ATE effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
library(CIMTx)
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_bart_ate(y = data$y, x = data$covariates , w = data$w, ndpost = 10)
```

---

`ce_estimate_bart_att` *This function implements the BART method when estimand is ATT. Please use our main function ce_estimate.R.*

---

## Description

This function implements the BART method when estimand is ATT. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_bart_att(
  y,
  x,
  w,
  discard = "No",
  ndpost = 1000,
  reference_trt,
  ...
)
```

## Arguments

| | |
|---|---|
| y | a numeric vector (0, 1) representing a binary outcome |
| x | a dataframe, including all the covariates but not treatments. |
| w | a numeric vector representing the treatment groups |
| discard | "No" or "Yes" indicating whether to use the discarding rules for the BART based method. The default is "No" |
| ndpost | number of posterior samples from BART |
| reference_trt | reference treatment group for ATT effect |
| ... | Other parameters that can be passed through the BART::pbart() function |

## Value

a list with w-1 elements for ATT effect; Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
library(CIMTx)
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
```

```
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_bart_att(y = data$y, x = data$covariates , w = data$w, ndpost = 10,reference_trt=1)
```

---

`ce_estimate_iptw_ate` *Inverse probability of treatment weighting (IPTW) for ATE estimation*

---

## Description

This function implements the IPTW method when estimand is ATE. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_iptw_ate(y, w, x, method, ...)
```

## Arguments

| | |
|---|---|
| `y` | numeric vector for the binary outcome |
| `w` | numeric vector for the treatment indicator |
| `x` | a dataframe, including all the covariates but not treatments |
| `method` | a character string. Users can selected from the following methods including "IPTW-Multinomial", "IPTW-GBM", "IPTW-SL" |
| `...` | Other parameters that can be passed through the twang::GBM() function |

## Value

a list with w*(w-1)/2 elements for ATE effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
```

```
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_iptw_ate(y = data$y, x = data$covariates,
w = data$w, ndpost=100, method = "IPTW-Multinomial")
```

---

`ce_estimate_iptw_att` *Inverse probability of treatment weighting (IPTW) for ATT estimation*

---

## Description

This function implements the IPTW method when estimand is ATT. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_iptw_att(y, x, w, method, reference_trt, ...)
```

## Arguments

| | |
|---|---|
| `y` | a numeric vector (0, 1) representing a binary outcome |
| `x` | a dataframe, including all the covariates but not treatments. |
| `w` | a numeric vector representing the treatment groups |
| `method` | a character string. Users can selected from the following methods including "IPTW-Multinomial", "IPTW-GBM", "IPTW-SL" |
| `reference_trt` | reference treatment group for ATT |
| `...` | Other parameters that can be passed through the twang::GBM() function |

## Value

a list with w-1 elements for ATT effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_iptw_att(y = data$y, x = data$covariates,
w = data$w, ndpost=100, method = "IPTW-Multinomial", reference_trt =1)
```

---

`ce_estimate_rams_ate` *Regression adjustment with multivariate spline of GPS (RAMS) for ATE estimation*

---

## Description

This function implements the RAMS method when estimand is ATE. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_rams_ate(y, w, x, method, ...)
```

## Arguments

| | |
|---|---|
| y | numeric vector for the binary outcome |
| w | numeric vector for the treatment indicator |
| x | dataframe including the treatment indicator and the covariates |
| method | a character string. Users can selected from the following methods including "RAMS-Multinomial", "RAMS-GBM", "RAMS-SL" |
| ... | Other paramters to be passed to twang::mnps() |

## Value

a list with w*(w-1)/2 elements for ATE effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_rams_ate(y = data$y, x = data$covariates ,
w = data$w, method = "RAMS-Multinomial")
```

`ce_estimate_rams_ate_boot` *Regression adjustment with multivariate spline of GPS (RAMS) for ATE estimation with bootstrapping*

## Description

This function implements bootstrapping for the RAMS method when estimand is ATE. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_rams_ate_boot(y, w, x, method, nboots, ...)
```

## Arguments

| | |
|---|---|
| `y` | numeric vector for the binary outcome |
| `w` | numeric vector for the treatment indicator |
| `x` | dataframe including the treatment indicator and the covariates |
| `method` | a character string. Users can selected from the following methods including "RAMS-Multinomial", "RAMS-GBM", "RAMS-SL" |
| `nboots` | a numeric value representing the number of bootstrap samples |
| `...` | Other paramters to be passed to twang::mnps() |

## Value

a list with w*(w-1)/2 elements for ATE effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
library(CIMTx)
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
"rnorm(300, 0, 0.5)",# x1
"rbeta(300, 2, .4)",   # x2
"runif(300, 0, 0.5)",# x3
"rweibull(300,1,2)",  # x4
"rbinom(300, 1, .4)"# x5
)
```

```
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_rams_ate_boot(y = data$y, x = data$covariates ,
w = data$w, method = "RAMS-Multinomial",nboots = 1)
```

---

`ce_estimate_rams_att` *Regression adjustment with multivariate spline of GPS (RAMS) for ATT estimation*

---

## Description

This function implements the RAMS method when estimand is ATT. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_rams_att(y, w, x, method, reference_trt, ...)
```

## Arguments

| | |
|---|---|
| `y` | numeric vector for the binary outcome |
| `w` | numeric vector for the treatment indicator |
| `x` | dataframe including the treatment indicator and the covariates |
| `method` | Methods to estimate GPS |
| `reference_trt` | Refercence group for ATT effect |
| `...` | Other paramters to be passed to twang::mnps() |

## Value

a list with w-1 elements for ATT effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_rams_att(y = data$y, x = data$covariates ,
w = data$w, method = "RAMS-Multinomial", reference_trt = 1)
```

---

`ce_estimate_ra_ate` *Regression Adjustment (RA) for ATE estimation*

---

## Description

This function implements the RA method when estimand is ATE. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_ra_ate(y, x, w, ndpost)
```

## Arguments

| | |
|---|---|
| y | numeric vector for the binary outcome |
| x | dataframe including the treatment indicator and the covariates |

| | |
|---|---|
| w | numeric vector for the treatment indicator |
| ndpost | number of independent simulation draws to create |

## Value

a list with w*(w-1)/2 elements for ATE effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
library(CIMTx)
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_ra_ate(y = data$y, x = data$covariates
, w = data$w, ndpost = 10)
```

---

`ce_estimate_ra_att` *Regression Adjustment (RA) for ATT estimation*

---

## Description

This function implements the RA method when estimand is ATE. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_ra_att(y, x, w, ndpost, reference_trt)
```

## Arguments

| | |
|---|---|
| y | numeric vector for the binary outcome |
| x | dataframe including the treatment indicator and the covariates |
| w | numeric vector for the treatment indicator |
| ndpost | number of independent simulation draws to create |
| reference_trt | Reference group for ATT |

## Value

a list with w-1 elements for ATT effect; Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_ra_att(y = data$y, x = data$covariates ,
w = data$w, reference_trt = 1, ndpost = 10)
```

`ce_estimate_tmle_ate` *Targeted Maximum Likelihood (TMLE) for ATE estimation*

## Description

This function implements the TMLE method when estimand is ATE. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_tmle_ate(y, w, x, SL.library, ...)
```

## Arguments

| | |
|---|---|
| `y` | numeric vector for the binary outcome |
| `w` | numeric vector for the treatment indicator |
| `x` | data frame containing the treatment indicator and covariates |
| `SL.library` | a character vector of prediction algorithms. A list of functions included in the SuperLearner package can be found with listWrappers() |
| `...` | Other arguments |

## Details

This function implements the TMLE method. Please use our main function causal_multi_treat.R.

## Value

a list with w*(w-1)/2 elements for ATE effect. Each element of the list contains the estimation for RD/RR/OR

## Examples

```
library(CIMTx)
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
```

```
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_tmle_ate(y = data$y, x = data$covariates ,
w = data$w, SL.library = c("SL.glm", "SL.mean"))
```

---

`ce_estimate_vm_att` *Vector Macthing (VM) for ATT estimation*

---

## Description

This function implements the VM method when estimand is ATT. Please use our main function ce_estimate.R.

## Usage

```
ce_estimate_vm_att(y, x, w, reference_trt, caliper, n_cluster)
```

## Arguments

| | |
|---|---|
| `y` | numeric vector for the binary outcome |
| `x` | dataframe including the treatment indicator and the covariates |
| `w` | numeric vector for the treatment indicator |
| `reference_trt` | reference_trt group for ATT |
| `caliper` | is a numeric value denoting the caliper which should be used when matching on the logit of GPS within each cluster formed by K-means clustering. The caliper is in standardized units. For example, caliper = 0.25 means that all matches greater than 0.25 standard deviations of the logit of GPS are dropped. The default value is 0.25 |
| `n_cluster` | a numeric value denoting the number of clusters to form using K means clustering on the logit of GPS. The default value is 5 |

## Value

a list with w-1 elements for ATT effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
ce_estimate_vm_att(y = data$y, x = data$covariates,
w = data$w,reference_trt = 1, caliper = 0.25, n_cluster = 5)
```

---

`covariate_overlap` *Covariate overlap plot*

---

## Description

This function plot the boxplots of the distributions of true GPS. Please use our data_sim.R to get the figure.

## Usage

```
covariate_overlap(treatment, prob)
```

## Arguments

| | |
|---|---|
| `treatment` | Treatment indicators |
| `prob` | Probability of receving the outcome |

## Value

a ggplot object

## Examples

```
probs_all <- matrix(runif(300,0,1), ncol = 3)
trt <- sample(c(1,2,3),100, replace = TRUE)
covariate_overlap(treatment = trt, prob = probs_all)
```

---

`data_sim` *Simulate data for binary outcome with multiple treatments*

---

## Description

This function simulate data for binary outcome with multiple treatments. Users can adjust the follwoing 7 design factors: (1) sample size, (2) ratio of units across treatment groups, (3) whether the treatment assignment model and the outcome generating model are linear or nonlinear, (4) whether the covariates that best predict the treatment also predict the outcome well, (5) whether the response surfaces are parallel across treatment groups, (6) outcome prevalence, and (7) degree of covariate overlap.

## Usage

```
data_sim(
  sample_size,
  n_trt,
  X,
  lp_y,
  nlp_y,
  align = TRUE,
  tau,
  delta,
  psi,
  lp_w,
  nlp_w
)
```

## Arguments

| | |
|---|---|
| `sample_size` | total number of units. |
| `n_trt` | the number of treatments |
| `X` | a vector of characters representing covariates, with each covariate being generated from the standard probability distributions in the stats package |
| `lp_y` | a vector of characters of length n_trt, representing the linear effects in the outcome generating model |

| | |
|---|---|
| nlp_y | a vector of characters of length n_trt, representing the nonlinear effects in the outcome generating model |
| align | logical,indicating whether the predictors in the treatment assignment model are the same as the predictors for the outcome generating model. The default is TRUE. If the argument is set to FALSE, users need to specify additional two arguments lp_w and nlp_w. |
| tau | a numeric vector of length n_trt inducing different outcome event probabilities across treatment groups |
| delta | a numeric vector of length n_trt-1 inducing different ratio of units across treatment groups. |
| psi | a numeric value for the parameter psi in the treatment assignment model, governing the sparsity of covariate overlap. |
| lp_w | is a vector of characters of length n_trt - 1, representing in the treatment assignment model |
| nlp_w | is a vector of characters of length n_trt - 1, representing in the treatment assignment model |

## Value

list with 7 elements for simulated data. It contains

| | |
|---|---|
| covariates: | X matrix |
| w: | treatment indicators |
| y: | observed binary outcomes |
| y_prev: | outcome prevalence rates |
| ratio_of_units: | the proportions of units in each treatment group |
| overlap_fig: | the visualization of covariate overlap via boxplots of the distributions of true GPS |
| Y_true_matrix: | simulated true outcome in each treatment group |

## Examples

```
library(CIMTx)
lp_w_all <-
 c(".4*x1 + .1*x2  - .1*x4 + .1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - .2 * x4 - .3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - .1*x4 - .2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(300, 0, 0.5)",# x1
 "rbeta(300, 2, .4)",   # x2
 "runif(300, 0, 0.5)",# x3
 "rweibull(300,1,2)",  # x4
```

```
 "rbinom(300, 1, .4)"# x5
)
set.seed(111111)
data <- data_sim(
 sample_size = 300,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(-1.5,0,1.5),
 delta = c(0.5,0.5),
 psi = 1
)
```

---

logit *Logit function*

---

## Description

Logit function

## Usage

```
logit(x)
```

## Arguments

| | |
|---|---|
| x | probability from 0 to 1 |

## Value

Logit of x

## Examples

```
logit(0.5)
```

## plot_boxplot *Boxplot for weight distribution*

### Description

This function make the boxplot plot for the weights estimated by different IPTW methods. The inputs of the function are from the output of ce_estimate.R function when the methods are "IPTW-Multinomial", "IPTW-GBM", "IPTW-SL".

### Usage

```
plot_boxplot(...)
```

### Arguments

| | |
|---|---|
| ... | Objects from IPTW related methods |

### Value

A ggplot figure

### Examples

```
iptw_object_example <- list(weight = rnorm(1000,1,1), method = "IPTW-SL")
plot_boxplot(iptw_object_example)
```

## plot_contour *Contour plot*

### Description

This function make the countor plot after the grid specification of sensitivity analysis. The input of the function is from the output of the sa.R function.

### Usage

```
plot_contour(sa_object, ATE = NULL, ATT = NULL)
```

### Arguments

| | |
|---|---|
| sa_object | Object from sa function |
| ATE | a character indicating the ATE effect to plot, eg, "1,3" or "2,3" |
| ATT | a character indicating the ATT effect to plot, eg, "1,3" or "1,2" |

## Value

A ggplot figure

## Examples

```
sa_object_example <- list(ATE13 = seq(0,1,length.out = 25), grid_index = c(4,5),
c_functions = data.frame(c4 = rep(seq(-0.6,0,0.15), each = 5),
c5 = rep(seq(0,0.6,0.15), 5)))
plot_contour(sa_object_example, ATE = "1,3")
```

---

# posterior_summary *Summarize posterior samples*

---

## Description

This function summarize posterior samples of RD, RR and OR. Please use our main function causal_multi_treat.R.

## Usage

```
posterior_summary(RD_est, RR_est, OR_est)
```

## Arguments

| | |
|---|---|
| RD_est | vector of estimation for RD |
| RR_est | vector of estimation for RR |
| OR_est | vector of estimation for OR |

## Value

a list with w-1 elements for ATT effect; a list with w*(w-1)/2 elements for ATE effect. Each element of the list contains the estimation, standard error, lower and upper 95% CI for RD/RR/OR

## Examples

```
library(CIMTx)
posterior_summary(RD_est = 1:10, RR_est = 11:20, OR_est = 1:10)
```

sa *Flexible Monte Carlo sensitivity analysis for unmeasured confounding*

## Description

Flexible Monte Carlo sensitivity analysis for unmeasured confounding

## Usage

```
sa(
  x,
  y,
  w,
  prior_c_function,
  M1,
  M2 = NULL,
  nCores,
  estimand,
  reference_trt,
  ...
)
```

## Arguments

| | |
|---|---|
| x | dataframe including the treatment indicator and the covariates |
| y | numeric vector for the binary outcome |
| w | numeric vector for the treatment indicator |
| prior_c_function | could be 1) a vector of characters indicating the prior distributions for the confounding functions. Each character contains the random number generation code from the standard probability distributions in the stats package. 2) a vector of characters including the grid specifications for the confounding functions. It should be used when users want to formulate the confounding functions as scalar values. 3)A matrix indicating the point mass prior for the confounding functions |
| M1 | a numeric value indicating the number of draws of the GPS from their posterior predictive distribution |
| M2 | a numeric value indicating the number of draws from the prior distributions of the confounding functions |
| nCores | number of cores to use for parallel computing |
| estimand | is a character string ("ATT", "ATE") representing the type of causal estimand |
| reference_trt | reference treatment group for the ATT effect |
| ... | Other parameters that can be passed to BART functions |

## Value

If prior_c_function include a grid specifications for the confounding functions, the output will be a list with 1) the drawn confounding function 2) w-1 causal estimand elements for ATT effect and w*(w-1)/2 causal estimand elements for ATE effect for each combination of the confounding functions. Otherwise, the output will be a data frame containing the estimation, standard error, lower and upper 95% CI for the causal estimand in terms of RD.

## Examples

```
lp_w_all <-
 c(".4*x1 + .1*x2  - 1.1*x4 + 1.1*x5",    # w = 1
   ".2 * x1 + .2 * x2  - 1.2 * x4 - 1.3 * x5")  # w = 2
nlp_w_all <-
 c("-.5*x1*x4  - .1*x2*x5", # w = 1
   "-.3*x1*x4 + .2*x2*x5")# w = 2
lp_y_all <- rep(".2*x1 + .3*x2 - .1*x3 - 1.1*x4 - 1.2*x5", 3)
nlp_y_all <- rep(".7*x1*x1  - .1*x2*x3", 3)
X_all <- c(
 "rnorm(100, 0, 0.5)",# x1
 "rbeta(100, 2, .4)",   # x2
 "runif(100, 0, 0.5)",# x3
 "rweibull(100,1,2)",  # x4
 "rbinom(100, 1, .4)"# x5
)
set.seed(1111)
data <- data_sim(
 sample_size = 100,
 n_trt = 3,
 X = X_all,
 lp_y = lp_y_all,
 nlp_y  = nlp_y_all,
 align = FALSE,
 lp_w = lp_w_all,
 nlp_w = nlp_w_all,
 tau = c(0.5,-0.5,0.5),
 delta = c(0.5,0.5),
 psi = 2
)
c_grid <- c(
 "runif(-0.6, 0)",# c(1,2)
 "runif(0, 0.6)",# c(2,1)
 "runif(-0.6, 0)", # c(2,3)
 "seq(-0.6, 0, by = 0.3)", # c(1,3)
 "seq(0, 0.6, by = 0.3)", # c(3,1)
 "runif(0, 0.6)" # c(3,2)
)
sensitivity_analysis_parallel_result <-
 sa(
   M1 = 1,
   x = data$covariates,
   y = data$y,
```

```
  w = data$w,
  prior_c_function = c_grid,
  nCores = 1,
  estimand = "ATE",
)
```

---

`trunc_fun` *Function to truncate weight*

---

## Description

Function to truncate weight

## Usage

```
trunc_fun(x, trim_perc = 0.05)
```

## Arguments

| | |
|---|---|
| `x` | weight |
| `trim_perc` | the percentile at which the inverse probability of treatment weights should be trimmed. |

## Value

Truncated weights

## Examples

```
trunc_fun(rnorm(1000,0,1),0.05)
```

# Index