Package: aihuman (via r-universe)

September 29, 2024

Type Package

Title Experimental Evaluation of Algorithm-Assisted Human Decision-Making

Version 0.1.0

Date 2023-02-22

Description Provides statistical methods for analyzing experimental evaluation of the causal impacts of algorithmic recommendations on human decisions developed by Imai, Jiang, Greiner, Halen, and Shin (2023) <doi:10.1093/jrsssa/qnad010>. The data used for this paper, and made available here, are interim, based on only half of the observations in the study and (for those observations) only half of the study follow-up period. We use them only to illustrate methods, not to draw substantive conclusions.

License GPL (>= 2)

URL https://github.com/sooahnshin/aihuman

BugReports https://github.com/sooahnshin/aihuman/issues

Encoding UTF-8

RoxygenNote 7.2.1

Imports Rcpp, coda, stats, magrittr, purrr, abind, foreach, parallel, doParallel, ggplot2, dplyr, tidyr, metR, MASS, lme4

LinkingTo Rcpp, RcppArmadillo, RcppEigen

Depends R (>= 2.10)

Suggests knitr, rmarkdown

VignetteBuilder knitr

LazyData true

NeedsCompilation yes

Author Sooahn Shin [aut, cre] (<https://orcid.org/0000-0001-6213-2197>), Zhichao Jiang [aut], Kosuke Imai [aut]

Maintainer Sooahn Shin <sooahnshin@g.harvard.edu>

Repository CRAN

Date/Publication 2023-03-02 06:50:02 UTC

Contents

---

aihuman-package *Experimental Evaluation of Algorithm-Assisted Human Decision-Making*

Description

Provides statistical methods for analyzing experimental evaluation of the causal impacts of algorithmic recommendations on human decisions developed by Imai, Jiang, Greiner, Halen, and Shin (2023) <doi:10.1093/jrsssa/qnad010>. The data used for this paper, and made available here, are interim, based on only half of the observations in the study and (for those observations) only half of the study follow-up period. We use them only to illustrate methods, not to draw substantive conclusions.

Package Content

Index of help topics:

```
APCEsummary             Summary of APCE
APCEsummaryipw          Summary of APCE for frequentist analysis
AiEvalmcmc              Gibbs sampler for the main analysis
BootstrapAPCEipw        Bootstrap for estimating variance of APCE
BootstrapAPCEipwRE      Bootstrap for estimating variance of APCE with
                        random effects
BootstrapAPCEipwREparallel
                        Bootstrap for estimating variance of APCE with
                        random effects
CalAPCE                 Calculate APCE
CalAPCEipw              Compute APCE using frequentist analysis
CalAPCEipwRE            Compute APCE using frequentist analysis with
                        random effects
CalAPCEparallel         Calculate APCE using parallel computing
CalDIM                  Calculate diff-in-means estimates
CalDIMsubgroup          Calculate diff-in-means estimates
CalDelta                Calculate the delta given the principal stratum
CalFairness             Calculate the principal fairness
CalOptimalDecision      Calculate optimal decision & utility
CalPS                   Calculate the proportion of principal strata
                        (R)
FTAdata                 Interim Dane data with failure to appear (FTA)
                        as an outcome
HearingDate             Interim court event hearing date
NCAdata                 Interim Dane data with new criminal activity
                        (NCA) as an outcome
NVCAdata                Interim Dane data with new violent criminal
```

	activity (NVCA) as an outcome
PSAdata	Interim Dane PSA data
PlotAPCE	Plot APCE
PlotDIMdecisions	Plot diff-in-means estimates
PlotDIMoutcomes	Plot diff-in-means estimates
PlotFairness	Plot the principal fairness
PlotOptimalDecision	Plot optimal decision
PlotPS	Plot the proportion of principal strata (R)
PlotSpilloverCRT	Plot conditional randomization test
PlotSpilloverCRTpower	Plot power analysis of conditional randomization test
PlotStackedBar	Stacked barplot for the distribution of the decision given psa
PlotStackedBarDMF	Stacked barplot for the distribution of the decision given DMF recommendation
PlotUtilityDiff	Plot utility difference
PlotUtilityDiffCI	Plot utility difference with 95 interval
SpilloverCRT	Conduct conditional randomization test
SpilloverCRTpower	Conduct power analysis of conditional randomization test
TestMonotonicity	Test monotonicity
TestMonotonicityRE	Test monotonicity with random effects
aihuman-package	Experimental Evaluation of Algorithm-Assisted Human Decision-Making
g_legend	Pulling ggplot legend
hearingdate_synth	Synthetic court event hearing date
psa_synth	Synthetic PSA data
synth	Synthetic data

Maintainer

NA

Author(s)

NA

`AiEvalmcmc` *Gibbs sampler for the main analysis*

Description

See Appendix S5 for more details.

Usage

```
AiEvalmcmc(
  data,
  rho = 0,
  Sigma0.beta.inv = NULL,
  Sigma0.alpha.inv = NULL,
  sigma0 = NULL,
  beta = NULL,
  alpha = NULL,
  theta = NULL,
  delta = NULL,
  n.mcmc = 5 * 10,
  verbose = FALSE,
  out.length = 10,
  beta.zx.off = FALSE,
  theta.z.off = FALSE
)
```

Arguments

data	A data.frame or matrix of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
rho	A sensitivity parameter. The default is 0 which implies the unconfoundedness assumption (Assumption 4).
Sigma0.beta.inv	Inverse of the prior covariance matrix of beta. The default is a diagonal matrix with 0.01 diagonal entries.
Sigma0.alpha.inv	Inverse of the prior covariance matrix of alpha. The default is a diagonal matrix with 0.01 diagonal entries.
sigma0	Prior variance of the cutoff points (theta and delta)
beta	Initial value for beta.
alpha	Initial value for alpha.
theta	Initial value for theta.
delta	Initial value for delta.
n.mcmc	The total number of MCMC iterations. The default is 50.
verbose	A logical argument specified to print the progress on the screen. The default is FALSE.
out.length	An integer to specify the progress on the screen. If verbose = TRUE, every out.length-th iteration is printed on the screen. The default is 10.
beta.zx.off	A logical argument specified to exclude the interaction terms (Z by X) from the model. The default is FALSE.
theta.z.off	A logical argument specified to set same cutoffs theta for treatment and control group. The default is FALSE.

Value

An object of class mcmc containing the posterior samples.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 2)
```

APCEsummary *Summary of APCE*

Description

Summary of average principal causal effects (APCE) with ordinal decision.

Usage

```
APCEsummary(apce.mcmc)
```

Arguments

apce.mcmc	APCE.mcmc array generated from CalAPCE or CalAPCEparallel.

Value

A data.frame that consists of mean and quantiles (2.5

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
sample_apce = CalAPCE(data = synth, mcmc.re = sample_mcmc, subgroup = subgroup_synth)
sample_apce_summary = APCEsummary(sample_apce[["APCE.mcmc"]])
```

`APCEsummaryipw` *Summary of APCE for frequentist analysis*

Description

Summary of average principal causal effects (APCE) with ordinal decision with frequentist results.

Usage

```
APCEsummaryipw(
  G1_est,
  G2_est,
  G3_est,
  G4_est,
  G5_est,
  G1_boot,
  G2_boot,
  G3_boot,
  G4_boot,
  G5_boot,
  name.group = c("Overall", "Female", "Male", "Non-white\nMale", "White\nMale")
)
```

Arguments

`G1_est`	List generated from `CalAPCEipw` for the first subgroup.
`G2_est`	List generated from `CalAPCEipw` for the second subgroup.
`G3_est`	List generated from `CalAPCEipw` for the third subgroup.
`G4_est`	List generated from `CalAPCEipw` for the fourth subgroup.
`G5_est`	List generated from `CalAPCEipw` for the fifth subgroup.
`G1_boot`	List generated from `BootstrapAPCEipw` for the first subgroup.
`G2_boot`	List generated from `BootstrapAPCEipw` for the second subgroup.
`G3_boot`	List generated from `BootstrapAPCEipw` for the third subgroup.
`G4_boot`	List generated from `BootstrapAPCEipw` for the fourth subgroup.
`G5_boot`	List generated from `BootstrapAPCEipw` for the fifth subgroup.
`name.group`	A list of character vectors for the label of five subgroups.

Value

A `data.frame` that consists of mean and quantiles (2.5

Examples

```
data(synth)
synth$SexWhite = synth$Sex * synth$White
freq_apce = CalAPCEipw(synth)
boot_apce = BootstrapAPCEipw(synth, rep = 10)
# subgroup analysis
data_s0 = subset(synth, synth$Sex==0,select=-c(Sex,SexWhite))
freq_s0 = CalAPCEipw(data_s0)
boot_s0 = BootstrapAPCEipw(data_s0, rep = 10)
data_s1 = subset(synth, synth$Sex==1,select=-c(Sex,SexWhite))
freq_s1 = CalAPCEipw(data_s1)
boot_s1 = BootstrapAPCEipw(data_s1, rep = 10)
data_s1w0 = subset(synth, synth$Sex==1&synth$White==0,select=-c(Sex,White,SexWhite))
freq_s1w0 = CalAPCEipw(data_s1w0)
boot_s1w0 = BootstrapAPCEipw(data_s1w0, rep = 10)
data_s1w1 = subset(synth, synth$Sex==1&synth$White==1,select=-c(Sex,White,SexWhite))
freq_s1w1 = CalAPCEipw(data_s1w1)
boot_s1w1 = BootstrapAPCEipw(data_s1w1, rep = 10)

freq_apce_summary <- APCEsummaryipw(freq_apce, freq_s0, freq_s1, freq_s1w0, freq_s1w1,
                                    boot_apce, boot_s0, boot_s1, boot_s1w0, boot_s1w0)
PlotAPCE(freq_apce_summary, y.max = 0.25, decision.labels = c("signature","small cash",
         "middle cash","large cash"), shape.values = c(16, 17, 15, 18),
         col.values = c("blue", "black", "red", "brown", "purple"), label = FALSE)
```

`BootstrapAPCEipw` *Bootstrap for estimating variance of APCE*

Description

Estimate variance of APCE for frequentist analysis using bootstrap. See S7 for more details.

Usage

```
BootstrapAPCEipw(data, rep = 1000)
```

Arguments

`data`	A `data.frame` or `matrix` of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
`rep`	Size of bootstrap

Value

An object of class `list` with the following elements:

`P.D1.boot`	An array with dimension rep by (k+1) by (k+2) for quantity P(D(1)=d| R=r), dimension 1 is rep (size of bootstrap), dimension 2 is (k+1) values of D from 0 to k, dimension 3 is (k+2) values of R from 0 to k+1.
`P.D0.boot`	An array with dimension rep by (k+1) by (k+2) for quantity P(D(0)=d| R=r).
`APCE.boot`	An array with dimension rep by (k+1) by (k+2) for quantity P(D(1)=d| R=r)-P(D(0)=d| R=r).
`P.R.boot`	An array with dimension rep by (k+2) for quantity P(R=r) for r from 0 to (k+1).
`alpha.boot`	An array with estimated alpha for each bootstrap.
`delta.boot`	An array with estimated delta for each bootstrap.

Examples

```
data(synth)
set.seed(123)
boot_apce = BootstrapAPCEipw(synth, rep = 100)
```

BootstrapAPCEipwRE *Bootstrap for estimating variance of APCE with random effects*

Description

Estimate variance of APCE for frequentist analysis with random effects using bootstrap. See S7 for more details.

Usage

```
BootstrapAPCEipwRE(data, rep = 1000, formula, CourtEvent_HearingDate, nAGQ = 1)
```

Arguments

`data`	A `data.frame` or `matrix` of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
`rep`	Size of bootstrap
`formula`	A formula of the model to fit.
`CourtEvent_HearingDate`	The court event hearing date.
`nAGQ`	Integer scalar - the number of points per axis for evaluating the adaptive Gauss-Hermite approximation to the log-likelihood. Defaults to 1, corresponding to the Laplace approximation.

Value

An object of class `list` with the following elements:

`P.D1.boot`	An array with dimension rep by (k+1) by (k+2) for quantity P(D(1)=d| R=r), dimension 1 is rep (size of bootstrap), dimension 2 is (k+1) values of D from 0 to k, dimension 3 is (k+2) values of R from 0 to k+1.
`P.D0.boot`	An array with dimension rep by (k+1) by (k+2) for quantity P(D(0)=d| R=r).
`APCE.boot`	An array with dimension rep by (k+1) by (k+2) for quantity P(D(1)=d| R=r)-P(D(0)=d| R=r).
`P.R.boot`	An array with dimension rep by (k+2) for quantity P(R=r) for r from 0 to (k+1).

Examples

```
data(synth)
data(hearingdate_synth)
synth$CourtEvent_HearingDate = hearingdate_synth
set.seed(123)
boot_apce_re = BootstrapAPCEipwRE(synth, rep = 10, formula = "Y ~ Sex + White + Age +
                                  CurrentViolentOffense + PendingChargeAtTimeOfOffense +
                                  PriorMisdemeanorConviction + PriorFelonyConviction +
                                PriorViolentConviction + (1|CourtEvent_HearingDate) + D",
                                  CourtEvent_HearingDate = hearingdate_synth)
```

BootstrapAPCEipwREparallel

Bootstrap for estimating variance of APCE with random effects

---

Description

Estimate variance of APCE for frequentist analysis with random effects using bootstrap. See S7 for more details.

Usage

```
BootstrapAPCEipwREparallel(data, rep = 1000, formula, nAGQ = 1, size = 5)
```

Arguments

`data`	A `data.frame` or `matrix` of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
`rep`	Size of bootstrap
`formula`	A formula of the model to fit.

nAGQ	Integer scalar - the number of points per axis for evaluating the adaptive Gauss-Hermite approximation to the log-likelihood. Defaults to 1, corresponding to the Laplace approximation.
size	The number of parallel computing. The default is 5.

Value

An object of class `list` with the following elements:

P.D1.boot	An array with dimension rep by (k+1) by (k+2) for quantity P(D(1)=d| R=r), dimension 1 is rep (size of bootstrap), dimension 2 is (k+1) values of D from 0 to k, dimension 3 is (k+2) values of R from 0 to k+1.
P.D0.boot	An array with dimension rep by (k+1) by (k+2) for quantity P(D(0)=d| R=r).
APCE.boot	An array with dimension rep by (k+1) by (k+2) for quantity P(D(1)=d| R=r)-P(D(0)=d| R=r).
P.R.boot	An array with dimension rep by (k+2) for quantity P(R=r) for r from 0 to (k+1).

Examples

```
data(synth)
data(hearingdate_synth)
synth$CourtEvent_HearingDate = hearingdate_synth
set.seed(123)
boot_apce_re = BootstrapAPCEipwREparallel(synth, rep = 10,
                                          formula = "Y ~ Sex + White + Age +
                                      CurrentViolentOffense + PendingChargeAtTimeOfOffense +
                                        PriorMisdemeanorConviction + PriorFelonyConviction +
                                       PriorViolentConviction + (1|CourtEvent_HearingDate) +
                                          D", size = 1) # adjust the size
```

CalAPCE *Calculate APCE*

Description

Calculate average principal causal effects (APCE) with ordinal decision. See Section 3.4 for more details.

Usage

```
CalAPCE(
  data,
  mcmc.re,
  subgroup,
  name.group = c("overall", "Sex0", "Sex1", "Sex1 White0", "Sex1 White1"),
  rho = 0,
```

```
  burnin = 0,
  out.length = 500,
  c0 = 0,
  c1 = 0,
  ZX = NULL,
  save.individual.optimal.decision = FALSE,
  parallel = FALSE,
  optimal.decision.only = FALSE,
  dmf = NULL,
  fair.dmf.only = FALSE
)
```

Arguments

data	A data.frame or matrix of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
mcmc.re	A mcmc object generated by AiEvalmcmc() function.
subgroup	A list of numeric vectors for the index of each of the five subgroups.
name.group	A list of character vectors for the label of five subgroups.
rho	A sensitivity parameter. The default is 0 which implies the unconfoundedness assumption (Assumption 4).
burnin	A proportion of burnin for the Markov chain. The default is 0.
out.length	An integer to specify the progress on the screen. Every out.length-th iteration is printed on the screen. The default is 500.
c0	The cost of an outcome. See Section 3.7 for more details. The default is 0.
c1	The cost of an unnecessarily harsh decision. See Section 3.7 for more details. The default is 0.
ZX	The data matrix for interaction terms. The default is the interaction between Z and all of the pre-treatment covariates (X).
save.individual.optimal.decision	A logical argument specified to save individual optimal decision rules. The default is FALSE.
parallel	A logical argument specifying whether parallel computing is conducted. Do not change this argument manually.
optimal.decision.only	A logical argument specified to compute only the optimal decision rule. The default is FALSE.
dmf	A numeric vector of binary DMF recommendations. If null, use judge's decisions (0 if the decision is 0 and 1 o.w; e.g., signature or cash bond).
fair.dmf.only	A logical argument specified to compute only the fairness of given DMF recommendations. The default is FALSE. Not used in the analysis for the JRSSA paper.

Value

An object of class `list` with the following elements:

`P.D1.mcmc` An array with dimension n.mcmc by 5 by (k+1) by (k+2) for quantity P(D(1)=d| R=r), dimension 1 is each posterior sample; dimension 2 is subgroup, dimension 3 is (k+1) values of D from 0 to k, dimension 4 is (k+2) values of R from 0 to k+1.

`P.D0.mcmc` An array with dimension n.mcmc by 5 by (k+1) by (k+2) for quantity P(D(0)=d| R=r).

`APCE.mcmc` An array with dimension n.mcmc by 5 by (k+1) by (k+2) for quantity P(D(1)=d| R=r)-P(D(0)=d| R=r).

`P.R.mcmc` An array with dimension n.mcmc by 5 by (k+2) for quantity P(R=r) for r from 0 to (k+1).

`Optimal.Z.mcmc` An array with dimension n.mcmc by 5 for the proportion of the cases where treatment (PSA provided) is optimal.

`Optimal.D.mcmc` An array with dimension n.mcmc by 5 by (k+1) for the proportion of optimal decision rule (average over observations). If `save.individual.optimal.decision = TRUE`, the dimension would be n by (k+1) (average over mcmc samples).

`P.DMF.mcmc` An array with dimension n.mcmc by 5 by (k+1) by (k+2) for the proportion of binary DMF recommendations. Not used in the analysis for the JRSSA paper.

`Utility.g_d.mcmc`
Included if `save.individual.optimal.decision = TRUE`. An array with dimension n for the individual utility of judge's decisions.

`Utility.g_dmf.mcmc`
Included if `save.individual.optimal.decision = TRUE`. An array with dimension n for the individual utility of DMF recommendation.

`Utility.diff.control.mcmc`
Included if `save.individual.optimal.decision = TRUE`. An array with dimension n.mcmc for estimated difference in utility between judge's decisions and DMF recommendation among control group.

`Utility.diff.treated.mcmc`
Included if `save.individual.optimal.decision = TRUE`. An array with dimension n.mcmc for estimated difference in utility between judge's decisions and DMF recommendation among treated group.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 2)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
sample_apce = CalAPCE(data = synth, mcmc.re = sample_mcmc, subgroup = subgroup_synth)
```

`CalAPCEipw` *Compute APCE using frequentist analysis*

Description

Estimate propensity score and use Hajek estimator to compute APCE. See S7 for more details.

Usage

```
CalAPCEipw(data)
```

Arguments

`data`	A `data.frame` or `matrix` of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.

Value

An object of class `list` with the following elements:

`P.D1`	An array with dimension (k+1) by (k+2) for quantity P(D(1)=d| R=r), dimension 1 is (k+1) values of D from 0 to k, dimension 2 is (k+2) values of R from 0 to k+1.
`P.D0`	An array with dimension (k+1) by (k+2) for quantity P(D(0)=d| R=r).
`APCE`	An array with dimension (k+1) by (k+2) for quantity P(D(1)=d| R=r)-P(D(0)=d| R=r).
`P.R`	An array with dimension (k+2) for quantity P(R=r) for r from 0 to (k+1).
`alpha`	An array with estimated alpha.
`delta`	An array with estimated delta.

Examples

```
data(synth)
freq_apce = CalAPCEipw(synth)
```

`CalAPCEipwRE` *Compute APCE using frequentist analysis with random effects*

Description

Estimate propensity score and use Hajek estimator to compute APCE. See S7 for more details.

Usage

```
CalAPCEipwRE(data, formula, nAGQ = 1)
```

Arguments

`data`	A `data.frame` or `matrix` of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
`formula`	A formula of the model to fit.
`nAGQ`	Integer scalar - the number of points per axis for evaluating the adaptive Gauss-Hermite approximation to the log-likelihood. Defaults to 1, corresponding to the Laplace approximation.

Value

An object of class `list` with the following elements:

`P.D1`	An array with dimension (k+1) by (k+2) for quantity P(D(1)=d| R=r), dimension 1 is (k+1) values of D from 0 to k, dimension 2 is (k+2) values of R from 0 to k+1.
`P.D0`	An array with dimension (k+1) by (k+2) for quantity P(D(0)=d| R=r).
`APCE`	An array with dimension (k+1) by (k+2) for quantity P(D(1)=d| R=r)-P(D(0)=d| R=r).
`P.R`	An array with dimension (k+2) for quantity P(R=r) for r from 0 to (k+1).
`alpha`	An array with estimated alpha.
`delta`	An array with estimated delta.

Examples

```
data(synth)
data(hearingdate_synth)
synth$CourtEvent_HearingDate = hearingdate_synth
freq_apce_re = CalAPCEipwRE(synth, formula = "Y ~ Sex + White + Age +
                            CurrentViolentOffense + PendingChargeAtTimeOfOffense +
                            PriorMisdemeanorConviction + PriorFelonyConviction +
                            PriorViolentConviction + (1|CourtEvent_HearingDate) + D")
```

`CalAPCEparallel` *Calculate APCE using parallel computing*

Description

Calculate average principal causal effects (APCE) with ordinal decision using parallel computing. See Section 3.4 for more details.

Usage

```
CalAPCEparallel(
  data,
  mcmc.re,
  subgroup,
  name.group = c("overall", "Sex0", "Sex1", "Sex1 White0", "Sex1 White1"),
  rho = 0,
  burnin = 0,
  out.length = 500,
  c0 = 0,
  c1 = 0,
  ZX = NULL,
  save.individual.optimal.decision = FALSE,
  optimal.decision.only = FALSE,
  dmf = NULL,
  fair.dmf.only = FALSE,
  size = 5
)
```

Arguments

`data`	A `data.frame` or `matrix` of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
`mcmc.re`	A `mcmc` object generated by `AiEvalmcmc()` function.
`subgroup`	A list of numeric vectors for the index of each of the five subgroups.
`name.group`	A list of character vectors for the label of five subgroups.
`rho`	A sensitivity parameter. The default is `0` which implies the unconfoundedness assumption (Assumption 4).
`burnin`	A proportion of burnin for the Markov chain. The default is `0`.
`out.length`	An integer to specify the progress on the screen. Every `out.length`-th iteration is printed on the screen. The default is `500`.
`c0`	The cost of an outcome. See Section 3.7 for more details. The default is `0`.
`c1`	The cost of an unnecessarily harsh decision. See Section 3.7 for more details. The default is `0`.

`ZX` The data matrix for interaction terms. The default is the interaction between Z and all of the pre-treatment covariates (X).

`save.individual.optimal.decision`
A logical argument specified to save individual optimal decision rules. The default is `FALSE`.

`optimal.decision.only`
A logical argument specified to compute only the optimal decision rule. The default is `FALSE`.

`dmf` A numeric vector of binary DMF recommendations. If `null`, use judge's decisions (0 if the decision is 0 and 1 o.w; e.g., signature or cash bond).

`fair.dmf.only` A logical argument specified to compute only the fairness of given DMF recommendations. The default is `FALSE`. Not used in the analysis for the JRSSA paper.

`size` The number of parallel computing. The default is `5`.

Value

An object of class `list` with the following elements:

`P.D1.mcmc` An array with dimension n.mcmc by 5 by (k+1) by (k+2) for quantity P(D(1)=d| R=r), dimension 1 is each posterior sample; dimension 2 is subgroup, dimension 3 is (k+1) values of D from 0 to k, dimension 4 is (k+2) values of R from 0 to k+1.

`P.D0.mcmc` An array with dimension n.mcmc by 5 by (k+1) by (k+2) for quantity P(D(0)=d| R=r).

`APCE.mcmc` An array with dimension n.mcmc by 5 by (k+1) by (k+2) for quantity P(D(1)=d| R=r)-P(D(0)=d| R=r).

`P.R.mcmc` An array with dimension n.mcmc by 5 by (k+2) for quantity P(R=r) for r from 0 to (k+1).

`Optimal.Z.mcmc` An array with dimension n.mcmc by 5 for the proportion of the cases where treatment (PSA provided) is optimal.

`Optimal.D.mcmc` An array with dimension n.mcmc by 5 by (k+1) for the proportion of optimal decision rule.

`P.DMF.mcmc` An array with dimension n.mcmc by 5 by (k+1) by (k+2) for the proportion of binary DMF recommendations. Not used in the analysis for the JRSSA paper.

`Utility.g_d.mcmc`
Included if `save.individual.optimal.decision = TRUE`. An array with dimension n for the individual utility of judge's decisions.

`Utility.g_dmf.mcmc`
Included if `save.individual.optimal.decision = TRUE`. An array with dimension n for the individual utility of DMF recommendation.

`Utility.diff.control.mcmc`
Included if `save.individual.optimal.decision = TRUE`. An array with dimension n.mcmc for estimated difference in utility between judge's decisions and DMF recommendation among control group.

`Utility.diff.treated.mcmc`
: Included if `save.individual.optimal.decision = TRUE`. An array with dimension n.mcmc for estimated difference in utility between judge's decisions and DMF recommendation among treated group.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
sample_apce = CalAPCEparallel(data = synth, mcmc.re = sample_mcmc,
                              subgroup = subgroup_synth,
                              size = 1) # adjust the size
```

`CalDelta` *Calculate the delta given the principal stratum*

Description

Calculate the maximal deviation of decisions probability among the distributions for different groups (delta) given the principal stratum (R).

Usage

```
CalDelta(r, k, pd0, pd1, attr)
```

Arguments

`r`	The given principal stratum.
`k`	The maximum decision (e.g., largest bail amount).
`pd0`	P.D0.mcmc array generated from `CalAPCE` or `CalAPCEparallel`.
`pd1`	P.D1.mcmc array generated from `CalAPCE` or `CalAPCEparallel`.
`attr`	The index of subgroups (within the output of CalAPCE/CalAPCEparallel) that corresponds to the protected attributes.

Value

A `data.frame` of the delta.

Examples

```
data(synth)
subgroup_synth = list(1:nrow(synth), which(synth$Sex==0), which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0), which(synth$Sex==1&synth$White==1))
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
sample_apce = CalAPCE(data = synth, mcmc.re = sample_mcmc, subgroup = subgroup_synth,
                      burnin = 0)
CalDelta(0, 3, sample_apce[["P.D0.mcmc"]], sample_apce[["P.D1.mcmc"]], c(2,3))
```

`CalDIM` *Calculate diff-in-means estimates*

Description

Calculate average causal effect based on diff-in-means estimator.

Usage

```
CalDIM(data)
```

Arguments

`data`	A `data.frame` of which columns includes a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y).

Value

A `data.frame` of diff-in-means estimates effect for each value of D and Y.

Examples

```
data(synth)
CalDIM(synth)
```

CalDIMsubgroup *Calculate diff-in-means estimates*

Description

Calculate average causal effect based on diff-in-means estimator.

Usage

```
CalDIMsubgroup(
  data,
  subgroup,
  name.group = c("Overall", "Female", "Male", "Non-white\nMale", "White\nMale")
)
```

Arguments

data	A data.frame of which columns includes a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y).
subgroup	A list of numeric vectors for the index of each of the five subgroups.
name.group	A character vector including the labels of five subgroups.

Value

A data.frame of diff-in-means estimates for each value of D and Y for each subgroup.

Examples

```
data(synth)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
CalDIMsubgroup(synth, subgroup = subgroup_synth)
```

CalFairness *Calculate the principal fairness*

Description

See Section 3.6 for more details.

Usage

```
CalFairness(apce, attr = c(2, 3))
```

Arguments

apce	The list generated from `CalAPCE` or `CalAPCEparallel`.
attr	The index of subgroups (within the output of CalAPCE/CalAPCEparallel) that corresponds to the protected attributes.

Value

A `data.frame` of the delta.

Examples

```
data(synth)
subgroup_synth = list(1:nrow(synth), which(synth$Sex==0), which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0), which(synth$Sex==1&synth$White==1))
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
sample_apce = CalAPCE(data = synth, mcmc.re = sample_mcmc, subgroup = subgroup_synth,
                      burnin = 0)
CalFairness(sample_apce)
```

CalOptimalDecision *Calculate optimal decision & utility*

Description

(1) Calculate optimal decision for each observation given each of c0 (cost of an outcome) and c1 (cost of an unnecessarily harsh decision) from the lists. (2) Calculate difference in the expected utility between binary version of judge's decisions and DMF recommendations given each of c0 (cost of an outcome) and c1 (cost of an unnecessarily harsh decision) from the lists.

Usage

```
CalOptimalDecision(
  data,
  mcmc.re,
  c0.ls,
  c1.ls,
  dmf = NULL,
  rho = 0,
  burnin = 0,
  out.length = 500,
  ZX = NULL,
  size = 5,
  include.utility.diff.mcmc = FALSE
)
```

Arguments

data	A data.frame or matrix of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
mcmc.re	A mcmc object generated by AiEvalmcmc() function.
c0.ls	The list of cost of an outcome. See Section 3.7 for more details.
c1.ls	The list of cost of an unnecessarily harsh decision. See Section 3.7 for more details.
dmf	A numeric vector of binary DMF recommendations. If null, use judge's decisions (0 if the decision is 0 and 1 o.w; e.g., signature or cash bond).
rho	A sensitivity parameter. The default is 0 which implies the unconfoundedness assumption (Assumption 4).
burnin	A proportion of burnin for the Markov chain. The default is 0.
out.length	An integer to specify the progress on the screen. Every out.length-th iteration is printed on the screen. The default is 500.
ZX	The data matrix for interaction terms. The default is the interaction between Z and all of the pre-treatment covariates (X).
size	The number of parallel computing. The default is 5.
include.utility.diff.mcmc	A logical argument specifying whether to save Utility.diff.control.mcmc and Utility.diff.treated.mcmc for Figure S17. The default is FALSE.

Value

A data.frame of (1) the probability that the optimal decision for each observation being d in 0,1,...,k, (2) expected utility of binary version of judge's decision (g_d), (3) expected utility of binary DMF recommendation, and (4) the difference between (2) and (3). If include.utility.diff.mcmc = TRUE, returns a list of such data.frame and a data.frame that includes the result for mean and quantile of Utility.diff.control.mcmc and Utility.diff.treated.mcmc across mcmc samples.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
sample_optd = CalOptimalDecision(data = synth, mcmc.re = sample_mcmc,
                                 c0.ls = seq(0,5,1), c1.ls = seq(0,5,1),
                                 size = 1) # adjust the size
```

CalPS *Calculate the proportion of principal strata (R)*

Description

Calculate the proportion of each principal stratum (R).

Usage

```
CalPS(
  p.r.mcmc,
  name.group = c("Overall", "Female", "Male", "Non-white\nMale", "White\nMale")
)
```

Arguments

p.r.mcmc	P.R.mcmc array generated from CalAPCE or CalAPCEparallel.
name.group	A character vector including the labels of five subgroups.

Value

A data.frame of the proportion of each principal stratum.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
sample_apce = CalAPCE(data = synth, mcmc.re = sample_mcmc,
                      subgroup = subgroup_synth)
CalPS(sample_apce[["P.R.mcmc"]])
```

FTAdata *Interim Dane data with failure to appear (FTA) as an outcome*

Description

An interim dataset containing pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The data used for the paper, and made available here, are interim, based on only half of the observations in the study and (for those observations) only half of the study follow-up period. We use them only to illustrate methods, not to draw substantive conclusions.

Usage

```
FTAdata
```

Format

A data frame with 1891 rows and 19 variables:

Z binary treatment

D ordinal decision

Y outcome

Sex male or female

White white or non-white

SexWhite the interaction between gender and race

Age age

PendingChargeAtTimeOfOffense binary variable for pending charge (felony, misdemeanor, or both) at the time of offense

NCorNonViolentMisdemeanorCharge binary variable for current non-violent felony charge

ViolentMisdemeanorCharge binary variable for current violent misdemeanor charge

ViolentFelonyCharge binary variable for current violent felony charge

NonViolentFelonyCharge binary variable for current non-violent felony charge

PriorMisdemeanorConviction binary variable for prior conviction of misdemeanor

PriorFelonyConviction binary variable for prior conviction of felony

PriorViolentConviction four-level ordinal variable for prior violent conviction

PriorSentenceToIncarceration binary variable for prior sentence to incarceration

PriorFTAInPastTwoYears three-level ordinal variable for FTAs from past two years

PriorFTAOlderThanTwoYears binary variable for FTAs from over two years ago

Staff_ReleaseRecommendation four-level ordinal variable for the DMF recommendation

---

g_legend *Pulling ggplot legend*

Description

Pulling ggplot legend

Usage

```
g_legend(p)
```

Arguments

p	A ggplot of which legend should be pulled.

Value

A ggplot legend.

HearingDate *Interim court event hearing date*

Description

An Interim Dane court event hearing date of Dane data in factor format. The data used for the paper, and made available here, are interim, based on only half of the observations in the study and (for those observations) only half of the study follow-up period. We use them only to illustrate methods, not to draw substantive conclusions.

Usage

```
HearingDate
```

Format

A date variable in factor format.

---

hearingdate_synth *Synthetic court event hearing date*

Description

A synthetic court event hearing date

Usage

```
hearingdate_synth
```

Format

A date variable.

NCAdata *Interim Dane data with new criminal activity (NCA) as an outcome*

Description

An interim dataset containing pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The data used for the paper, and made available here, are interim, based on only half of the observations in the study and (for those observations) only half of the study follow-up period. We use them only to illustrate methods, not to draw substantive conclusions.

Usage

```
NCAdata
```

Format

A data frame with 1891 rows and 19 variables:

Z binary treatment

D ordinal decision

Y outcome

Sex male or female

White white or non-white

SexWhite the interaction between gender and race

Age age

PendingChargeAtTimeOfOffense binary variable for pending charge (felony, misdemeanor, or both) at the time of offense

NCorNonViolentMisdemeanorCharge binary variable for current non-violent felony charge

ViolentMisdemeanorCharge binary variable for current violent misdemeanor charge

ViolentFelonyCharge binary variable for current violent felony charge

NonViolentFelonyCharge binary variable for current non-violent felony charge

PriorMisdemeanorConviction binary variable for prior conviction of misdemeanor

PriorFelonyConviction binary variable for prior conviction of felony

PriorViolentConviction four-level ordinal variable for prior violent conviction

PriorSentenceToIncarceration binary variable for prior sentence to incarceration

PriorFTAInPastTwoYears three-level ordinal variable for FTAs from past two years

PriorFTAOlderThanTwoYears binary variable for FTAs from over two years ago

Staff_ReleaseRecommendation four-level ordinal variable for the DMF recommendation

NVCAdata *Interim Dane data with new violent criminal activity (NVCA) as an outcome*

Description

An interim dataset containing pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The data used for the paper, and made available here, are interim, based on only half of the observations in the study and (for those observations) only half of the study follow-up period. We use them only to illustrate methods, not to draw substantive conclusions.

Usage

```
NVCAdata
```

Format

A data frame with 1891 rows and 19 variables:

Z binary treatment

D ordinal decision

Y outcome

Sex male or female

White white or non-white

SexWhite the interaction between gender and race

Age age

PendingChargeAtTimeOfOffense binary variable for pending charge (felony, misdemeanor, or both) at the time of offense

NCorNonViolentMisdemeanorCharge binary variable for current non-violent felony charge

ViolentMisdemeanorCharge binary variable for current violent misdemeanor charge

ViolentFelonyCharge binary variable for current violent felony charge

NonViolentFelonyCharge binary variable for current non-violent felony charge

PriorMisdemeanorConviction binary variable for prior conviction of misdemeanor

PriorFelonyConviction binary variable for prior conviction of felony

PriorViolentConviction four-level ordinal variable for prior violent conviction

PriorSentenceToIncarceration binary variable for prior sentence to incarceration

PriorFTAInPastTwoYears three-level ordinal variable for FTAs from past two years

PriorFTAOlderThanTwoYears binary variable for FTAs from over two years ago

Staff_ReleaseRecommendation four-level ordinal variable for the DMF recommendation

PlotAPCE *Plot APCE*

Description

See Figure 4 for example.

Usage

```
PlotAPCE(
  res,
  y.max = 0.1,
  decision.labels = c("signature bond", "small cash bond", "large cash bond"),
  shape.values = c(16, 17, 15),
  col.values = c("blue", "black", "red", "brown"),
  label = TRUE,
  r.labels = c("safe", "easily\npreventable", "prevent-\nable", "risky\n"),
  label.position = c("top", "top", "top", "top"),
  top.margin = 0.01,
  bottom.margin = 0.01,
  name.group = c("Overall", "Female", "Male", "Non-white\nMale", "White\nMale"),
  label.size = 4
)
```

Arguments

res	A data.frame generated with APCEsummary().
y.max	Maximum value of y-axis.
decision.labels	Labels of decisions (D).
shape.values	Shape of point for each decisions.
col.values	Color of point for each principal stratum.
label	A logical argument whether to specify label of each principal stratum. The default is TRUE.
r.labels	Label of each principal stratum.
label.position	The position of labels.
top.margin	Top margin of labels.
bottom.margin	Bottom margin of labels.
name.group	A character vector including the labels of five subgroups.
label.size	Size of label.

Value

A ggplot.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
sample_apce = CalAPCE(data = synth, mcmc.re = sample_mcmc,
                      subgroup = subgroup_synth)
sample_apce_summary = APCEsummary(sample_apce[["APCE.mcmc"]])
PlotAPCE(sample_apce_summary, y.max = 0.25, decision.labels = c("signature","small cash",
         "middle cash","large cash"), shape.values = c(16, 17, 15, 18),
         col.values = c("blue", "black", "red", "brown", "purple"), label = FALSE)
```

PlotDIMdecisions *Plot diff-in-means estimates*

Description

See Figure 2 for example.

Usage

```
PlotDIMdecisions(
  res,
  y.max = 0.2,
 decision.labels = c("signature bond  ", "small cash bond  ", "large cash bond"),
  col.values = c("grey60", "grey30", "grey6"),
  shape.values = c(16, 17, 15)
)
```

Arguments

`res`	A `data.frame` generated with `CalDIMsubgroup`.
`y.max`	Maximum value of y-axis.
`decision.labels`	Labels of decisions (D).
`col.values`	Color of point for each decisions.
`shape.values`	Shape of point for each decisions.

Value

A ggplot.

Examples

```
data(synth)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
res_dec = CalDIMsubgroup(synth, subgroup = subgroup_synth)
PlotDIMdecisions(res_dec, decision.labels = c("signature","small cash","middle cash","large cash"),
                 col.values = c("grey60", "grey30", "grey6", "grey1"),
                 shape.values = c(16, 17, 15, 18))
```

`PlotDIMoutcomes` *Plot diff-in-means estimates*

Description

See Figure 2 for example.

Usage

```
PlotDIMoutcomes(
  res.fta,
  res.nca,
  res.nvca,
  label.position = c("top", "top", "top"),
  top.margin = 0.01,
  bottom.margin = 0.01,
  y.max = 0.2,
  label.size = 7,
  name.group = c("Overall", "Female", "Male", "Non-white\nMale", "White\nMale")
)
```

Arguments

`res.fta`	A `data.frame` generated with `CalDIMsubgroup` with Y = FTA.
`res.nca`	A `data.frame` generated with `CalDIMsubgroup` with Y = NCA.
`res.nvca`	A `data.frame` generated with `CalDIMsubgroup` with Y = NVCA.
`label.position`	The position of labels.
`top.margin`	Top margin of labels.
`bottom.margin`	Bottom margin of labels.
`y.max`	Maximum value of y-axis.
`label.size`	Size of label.
`name.group`	A character vector including the labels of five subgroups.

Value

A ggplot.

Examples

```
data(synth)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
synth_fta <- synth_nca <- synth_nvca <- synth
set.seed(123)
synth_fta$Y <- sample(0:1, 1000, replace = TRUE)
synth_nca$Y <- sample(0:1, 1000, replace = TRUE)
synth_nvca$Y <- sample(0:1, 1000, replace = TRUE)
res_fta = CalDIMsubgroup(synth_fta, subgroup = subgroup_synth)
res_nca = CalDIMsubgroup(synth_nca, subgroup = subgroup_synth)
res_nvca = CalDIMsubgroup(synth_nvca, subgroup = subgroup_synth)
PlotDIMoutcomes(res_fta, res_nca, res_nvca)
```

`PlotFairness` *Plot the principal fairness*

Description

See Figure 5 for example.

Usage

```
PlotFairness(
  res,
  top.margin = 0.01,
  y.max = 0.2,
  y.min = -0.1,
  r.labels = c("Safe", "Easily\nPreventable", "Preventable", "Risky"),
  label = TRUE
)
```

Arguments

`res`	The data frame generated from `CalFairness`.
`top.margin`	The index of subgroups (within the output of CalAPCE/CalAPCEparallel) that corresponds to the protected attributes.
`y.max`	Maximum value of y-axis.
`y.min`	Minimum value of y-axis.
`r.labels`	Label of each principal stratum.
`label`	A logical argument whether to specify label.

Value

A `data.frame` of the delta.

Examples

```
data(synth)
subgroup_synth = list(1:nrow(synth), which(synth$Sex==0), which(synth$Sex==1),
                 which(synth$Sex==1&synth$White==0), which(synth$Sex==1&synth$White==1))
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
sample_apce = CalAPCE(data = synth, mcmc.re = sample_mcmc, subgroup = subgroup_synth,
                      burnin = 0)
sample_fair = CalFairness(sample_apce)
PlotFairness(sample_fair, y.max = 0.4, y.min = -0.4, r.labels = c("Safe", "Preventable 1",
             "Preventable 2", "Preventable 3", "Risky"))
```

PlotOptimalDecision *Plot optimal decision*

Description

See Figure 6 for example.

Usage

```
PlotOptimalDecision(res, colname.d, idx = NULL)
```

Arguments

res	The data frame generated from CalOptimalDecision.
colname.d	The column name of decision to be plotted.
idx	The row index of observations to be included. The default is all the observations from the data.

Value

A ggplot.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
sample_optd = CalOptimalDecision(data = synth, mcmc.re = sample_mcmc,
                                 c0.ls = seq(0,5,1), c1.ls = seq(0,5,1),
                                 size = 1) # adjust the size
sample_optd$cash = sample_optd$d1 + sample_optd$d2 + sample_optd$d3
PlotOptimalDecision(sample_optd, "cash")
```

PlotPS *Plot the proportion of principal strata (R)*

Description

See Figure 3 for example.

Usage

```
PlotPS(
  res,
  y.min = 0,
  y.max = 0.75,
  col.values = c("blue", "black", "red", "brown"),
  label = TRUE,
  r.labels = c("safe", " easily             \n preventable    ",
    "\n          preventable\n", "  risky"),
  label.position = c("top", "top", "top", "bottom"),
  top.margin = 0.02,
  bottom.margin = 0.02,
  label.size = 6.5
)
```

Arguments

res	A data.frame generated with CalPS.
y.min	Minimum value of y-axis.
y.max	Maximum value of y-axis.
col.values	Color of point for each principal stratum.
label	A logical argument whether to specify label of each principal stratum. The default is TRUE.
r.labels	Label of each principal stratum.
label.position	The position of labels.
top.margin	Top margin of labels.
bottom.margin	Bottom margin of labels.
label.size	Size of label.

Value

A ggplot.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
subgroup_synth = list(1:nrow(synth),which(synth$Sex==0),which(synth$Sex==1),
                  which(synth$Sex==1&synth$White==0),which(synth$Sex==1&synth$White==1))
sample_apce = CalAPCE(data = synth, mcmc.re = sample_mcmc,
                      subgroup = subgroup_synth)
sample_ps = CalPS(sample_apce[["P.R.mcmc"]])
PlotPS(sample_ps, col.values = c("blue", "black", "red", "brown", "purple"), label = FALSE)
```

`PlotSpilloverCRT` *Plot conditional randomization test*

Description

See Figure S8 for example.

Usage

```
PlotSpilloverCRT(res)
```

Arguments

`res`	A `list` generated with `SpilloverCRT`.

Value

A ggplot

Examples

```
data(synth)
data(hearingdate_synth)
crt <- SpilloverCRT(D = synth$D, Z = synth$Z, CourtEvent_HearingDate = hearingdate_synth)
PlotSpilloverCRT(crt)
```

PlotSpilloverCRTpower *Plot power analysis of conditional randomization test*

Description

See Figure S9 for example.

Usage

```
PlotSpilloverCRTpower(res)
```

Arguments

res	A data.frame generated with SpilloverCRTpower.

Value

A ggplot

Examples

```
data(synth)
data(hearingdate_synth)
crt_power <- SpilloverCRTpower(D = synth$D, Z = synth$Z,
                               CourtEvent_HearingDate = hearingdate_synth,
                               size = 1) # adjust the size
PlotSpilloverCRTpower(crt_power)
```

PlotStackedBar *Stacked barplot for the distribution of the decision given psa*

Description

See Figure 1 for example.

Usage

```
PlotStackedBar(
  data,
  fta.label = "FTAScore",
  nca.label = "NCAScore",
  nvca.label = "NVCAFlag",
  d.colors = c("grey60", "grey30", "grey10"),
  d.labels = c("signature bond", "small cash bond", "large cash bond"),
  legend.position = "none"
)
```

Arguments

data	A data.frame of which columns includes an ordinal decision (D), and psa variables (fta, nca, and nvca).
fta.label	Column name of fta score in the data. The default is "FTAScore".
nca.label	Column name of nca score in the data. The default is "NCAScore".
nvca.label	Column name of nvca score in the data. The default is "NVCAFlag".
d.colors	The color of each decision.
d.labels	The label of each decision.
legend.position	The position of legend. The default is "none".

Value

A list of three ggplots.

Examples

```
data(psa_synth)
# Control group (PSA not provided)
PlotStackedBar(psa_synth[psa_synth$Z == 0, ], d.colors = c("grey80", "grey60",
               "grey30", "grey10"), d.labels = c("signature","small",
               "middle","large"))
# Treated group (PSA provided)
PlotStackedBar(psa_synth[psa_synth$Z == 0, ], d.colors = c("grey80", "grey60",
               "grey30", "grey10"), d.labels = c("signature","small",
               "middle","large"))
```

PlotStackedBarDMF *Stacked barplot for the distribution of the decision given DMF recommendation*

Description

See Figure 1 for example.

Usage

```
PlotStackedBarDMF(
  data,
  dmf.label = "dmf",
  dmf.category = NULL,
  d.colors = c("grey60", "grey30", "grey10"),
  d.labels = c("signature bond", "small cash bond", "large cash bond"),
  legend.position = "none"
)
```

Arguments

data	A data.frame of which columns includes a binary treatment (Z; PSA provision), an ordinal decision (D), and DMF recommendation.
dmf.label	Column name of DMF recommendation in the data. The default is "dmf".
dmf.category	The name of each category of DMF recommendation.
d.colors	The color of each decision.
d.labels	The label of each decision.
legend.position	The position of legend. The default is "none".

Value

A list of three ggplots.

Examples

```
data(psa_synth)
PlotStackedBarDMF(psa_synth, dmf.label = "DMF", d.colors = c("grey80",
                  "grey60", "grey30", "grey10"), d.labels = c("signature",
                  "small","middle","large"))
```

PlotUtilityDiff *Plot utility difference*

Description

See Figure 7 for example.

Usage

```
PlotUtilityDiff(res, idx = NULL)
```

Arguments

res	The data frame generated from CalUtilityDiff.
idx	The row index of observations to be included. The default is all the observations from the data.

Value

A ggplot.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
synth_dmf = sample(0:1, nrow(synth), replace = TRUE) # random dmf recommendation
sample_utility = CalOptimalDecision(data = synth, mcmc.re = sample_mcmc,
                                    c0.ls = seq(0,5,1), c1.ls = seq(0,5,1),
                                    dmf = synth_dmf, size = 1) # adjust the size
PlotUtilityDiff(sample_utility)
```

PlotUtilityDiffCI *Plot utility difference with 95% confidence interval*

Description

See Figure S17 for example.

Usage

```
PlotUtilityDiffCI(res)
```

Arguments

res	The second data frame (res.mcmc) generated from `CalUtilityDiff(include.utility.diff.mcmc = TRUE)`.

Value

A ggplot.

Examples

```
data(synth)
sample_mcmc = AiEvalmcmc(data = synth, n.mcmc = 10)
synth_dmf = sample(0:1, nrow(synth), replace = TRUE) # random dmf recommendation
sample_utility = CalOptimalDecision(data = synth, mcmc.re = sample_mcmc,
                                    c0.ls = seq(0,5,1), c1.ls = seq(0,5,1),
                                    dmf = synth_dmf, size = 1, # adjust the size
                                    include.utility.diff.mcmc = TRUE)
PlotUtilityDiffCI(sample_utility$res.mcmc)
```

PSAdata *Interim Dane PSA data*

Description

An interim dataset containing a binary treatment (Z), ordinal decision (D), three PSA variables (FTAScore, NCAScore, and NVCAFlag), DMF recommendation, and two pre-treatment covariates (binary indicator for gender; binary indicator for race). The data used for the paper, and made available here, are interim, based on only half of the observations in the study and (for those observations) only half of the study follow-up period. We use them only to illustrate methods, not to draw substantive conclusions.

Usage

```
PSAdata
```

Format

A data frame with 1891 rows and 7 variables:

Z binary treatment

D ordinal decision

FTAScore FTA score

NCAScore NCA score

NVCAFlag NVCA flag

DMF DMF recommendation

Sex male or female

White white or non-white

psa_synth *Synthetic PSA data*

Description

A synthetic dataset containing a binary treatment (Z), ordinal decision (D), three PSA variables (FTAScore, NCAScore, and NVCAFlag), and DMF recommendation.

Usage

```
psa_synth
```

Format

A data frame with 1000 rows and 4 variables:

Z binary treatment

D ordinal decision

FTAScore FTA score

NCAScore NCA score

NVCAFlag NVCA flag

DMF DMF recommendation

SpilloverCRT *Conduct conditional randomization test*

Description

See S3.1 for more details.

Usage

```
SpilloverCRT(D, Z, CourtEvent_HearingDate, n = 100, seed.number = 12345)
```

Arguments

D	A numeric vector of judge's decision.
Z	A numeric vector of treatment variable.
CourtEvent_HearingDate	The court event hearing date.
n	Number of permutations.
seed.number	An integer for random number generator.

Value

A `list` of the observed and permuted test statistics and its p-value.

Examples

```
data(synth)
data(hearingdate_synth)
crt <- SpilloverCRT(D = synth$D, Z = synth$Z, CourtEvent_HearingDate = hearingdate_synth)
```

SpilloverCRTpower *Conduct power analysis of conditional randomization test*

Description

See S3.2 for more details.

Usage

```
SpilloverCRTpower(
  D,
  Z,
  CourtEvent_HearingDate,
  n = 4,
  m = 4,
  size = 2,
  cand_omegaZtilde = seq(-1.5, 1.5, by = 0.5)
)
```

Arguments

D	A numeric vector of judge's decision.
Z	A numeric vector of treatment variable.
CourtEvent_HearingDate	The court event hearing date.
n	Number of permutations.
m	Number of permutations.
size	The number of parallel computing. The default is 2.
cand_omegaZtilde	Candidate values

Value

A data.frame of the result of power analysis.

Examples

```
data(synth)
data(hearingdate_synth)
crt_power <- SpilloverCRTpower(D = synth$D, Z = synth$Z,
                               CourtEvent_HearingDate = hearingdate_synth,
                               size = 1) # adjust the size
```

synth *Synthetic data*

Description

A synthetic dataset containing pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y).

Usage

```
synth
```

Format

A data frame with 1000 rows and 11 variables:

Z binary treatment

D ordinal decision

Y outcome

Sex male or female

White white or non-white

Age age

CurrentViolentOffense binary variable for current violent offense

PendingChargeAtTimeOfOffense binary variable for pending charge (felony, misdemeanor, or both) at the time of offense

PriorMisdemeanorConviction binary variable for prior conviction of misdemeanor

PriorFelonyConviction binary variable for prior conviction of felony

PriorViolentConviction four-level ordinal variable for prior violent conviction

TestMonotonicity *Test monotonicity*

Description

Test monotonicity using frequentist analysis

Usage

```
TestMonotonicity(data)
```

Arguments

data	A data.frame or matrix of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.

Value

Message indicating whether the monotonicity assumption holds.

Examples

```
data(synth)
TestMonotonicity(synth)
```

TestMonotonicityRE *Test monotonicity with random effects*

Description

Test monotonicity using frequentist analysis with random effects for the hearing date of the case.

Usage

```
TestMonotonicityRE(data, formula)
```

Arguments

data	A data.frame or matrix of which columns consists of pre-treatment covariates, a binary treatment (Z), an ordinal decision (D), and an outcome variable (Y). The column names of the latter three should be specified as "Z", "D", and "Y" respectively.
formula	A formula of the model to fit.

Value

Message indicating whether the monotonicity assumption holds.

Examples

```
data(synth)
data(hearingdate_synth)
synth$CourtEvent_HearingDate = hearingdate_synth
TestMonotonicityRE(synth, formula = "Y ~ Sex + White + Age +
                   CurrentViolentOffense + PendingChargeAtTimeOfOffense +
                   PriorMisdemeanorConviction + PriorFelonyConviction +
                   PriorViolentConviction + (1|CourtEvent_HearingDate) + D")
```

Index