Package 'sate'

February 6, 2025

Type Package

Title Scientific Analysis of Trial Errors (SATE)

Version 2.3.0

Description Bundles functions used to analyze the harmfulness of trial errors in criminal trials. Functions in the Scientific Analysis of Trial Errors ('SATE') package help users estimate the probability that a jury will find a defendant guilty given jurors' preferences for a guilty verdict and the uncertainty of that estimate. Users can also compare actual and hypothetical trial conditions to conduct harmful error analysis. The relationship between individual jurors' verdict preferences and the probability that a jury returns a guilty verdict has been studied by Davis (1973) <doi:10.1037/h0033951>; MacCoun & Kerr (1988) <doi:10.1037/0022-3514.54.1.21>,
and Devine et el. (2001) <doi:10.1037/1076-8971.7.3.622>, among others.

License CC0

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports stats, ellipse, graphics, MASS, survey

RoxygenNote 7.3.2

NeedsCompilation no

Author Barry Edwards [aut, cre]

Maintainer Barry Edwards <bce@uga.edu>

Repository CRAN

Date/Publication 2025-02-06 17:40:02 UTC

Contents

`as.jury.point` *Calculates probability a jury will find defendant guilty based on juror preferences*

Description

Calculates the probability that jury of size jury_n finds defendant guilty given on preferences of jury pool (inputted as sample_pg). Does not estimate uncertainty (use as.jury.stats function for inferential statistics).

Usage

```
as.jury.point(
  sample_pg,
  jury_n = 12,
  pstrikes = 0,
  dstrikes = 0,
  accuracy = 0.15
)
```

Arguments

`sample_pg`	Proportion of jurors who favor a guilty verdict. Can be a single number between 0 and 1, or a vector of such numbers.
`jury_n`	Size of the jury (i.e. 6, 8, or 12); default value is 12.
`pstrikes`	Number of peremptory strikes by prosecution; default value is 0.
`dstrikes`	Number of peremptory strikes by defendant; default value is 0.
`accuracy`	Accuracy of parties' peremptory strikes; a number between 0 and 1; default value is .15.

Value

Returns the probability jury finds defendant guilty (if sample_pg is a single number) or vector of such probabilities (if sample_pg is a vector).

Examples

```
library(sate)
as.jury.point(sample_pg = .50)

as.jury.point(sample_pg = 10/12)
```

`as.jury.stats` *Calculates probability a jury will find defendant guilty based on juror preferences, with standard error and confidence interval*

Description

Calculates probability jury finds defendant guilty based on verdicts preferences of jury pool. Also reports standard error and confidence interval of estimate (use as.jury.point function for estimate only).

Usage

```
as.jury.stats(
  sample_pg,
  sample_n,
  jury_n = 12,
  pstrikes = 0,
  dstrikes = 0,
  accuracy = 0.15,
  digits = 3
)
```

Arguments

`sample_pg`	Proportion of jurors who favor a guilty verdict; a number between 0 and 1.
`sample_n`	Size of sample used to estimate sample_pg.
`jury_n`	Size of the jury (i.e. 6, 8, or 12); default value is 12.
`pstrikes`	Number of peremptory strikes by prosecution; default value is 0.
`dstrikes`	Number of peremptory strikes by defendant; default value is 0.
`accuracy`	Accuracy of parties' peremptory strikes; a number between 0 and 1; default value is .15.
`digits`	Number of digits to report after decimal places; default value is 3.

Value

Returns the probability jury finds defendant guilty.

Examples

```
library(sate)
as.jury.stats(sample_pg=.50, sample_n=830)

as.jury.stats(sample_pg=10/12, sample_n=295)
```

`basic.plot.grid` *Creates the shell of a plot showing relationship between jury pool preferences and jury verdict probabilities*

Description

Creates the shell of a plot showing relationship between jury pool preferences and jury verdict probabilities, optional argument to modify main, xlab, and ylab labels, includes grid lines.

Usage

```
basic.plot.grid(main, xlab, ylab)
```

Arguments

`main`	Main title for plot (optional), default is "Jurors' Verdict Preferences, P(g)".
`xlab`	X-axis label for plot (optional), default is "Jury Verdict Probabilities, P(G)".
`ylab`	Main title for plot (optional), default is no main title.

Value

No return

Examples

```
library(sate)
basic.plot.grid()

basic.plot.grid(main="Death Sentencing Analysis", xlab="Jurors' Sentencing Preferences, P(d)",
                ylab="Jury Verdict Probabilities, P(D)")
```

`compact_harm_plot` *Creates the shell of a plot used to display estimate of harm relative to harm threshold*

Description

Creates the shell of a plot used for compact display estimate of harm estimate relative to harm thresholds.

Usage

```
compact_harm_plot()
```

Value

No return

Examples

```
library(sate)
compact_harm_plot()
```

`compare.juror.stats` *Estimates juror-level differences based on sample statistics (from survey)*

Description

Calculates juror-level statistics and differences based on sample statistics (from a survey) supplied by user.

Usage

```
compare.juror.stats(pg_actual, n_actual, pg_hypo, n_hypo, digits = 3)
```

Arguments

`pg_actual`	The proportion of jurors who favor a guilty verdict in the actual trial condition (the trial with error).
`n_actual`	The size of the sample used to estimate pg_actual.
`pg_hypo`	The proportion of jurors who favor a guilty verdict in the hypothetical trial condition (the fair trial without error).
`n_hypo`	The size of the sample used to estimate pg_hypo.
`digits`	Number of digits to report after decimal places; default value is 3.

Value

Returns a list of juror-level statistics to assess the effect of a trial error or omission on juror preferences. Returned list includes statistics for the actual trial, the hypothetical trial, and the difference between them.

Examples

```
library(sate)
compare.juror.stats(pg_actual=.70, n_actual=400, pg_hypo=.60, n_hypo=450)

compare.juror.stats(pg_actual=.75, n_actual=450, pg_hypo=.65, n_hypo=350)
```

---

compare.jury.stats *Estimates jury-level differences based on juror-level statistics with inferential statistics*

Description

Calculates jury-level statistics and differences based on juror-level statistics supplied by user.

Usage

```
compare.jury.stats(
  pg_actual,
  n_actual,
  pg_hypo,
  n_hypo,
  jury_n = 12,
  pstrikes = 0,
  dstrikes = 0,
  accuracy = 0.15,
  digits = 3
)
```

Arguments

pg_actual	The proportion of jurors who favor a guilty verdict in the actual trial condition (the trial with error).
n_actual	The size of the sample used to estimate pg_actual.
pg_hypo	The proportion of jurors who favor a guilty verdict in the hypothetical trial condition (the fair trial without error).
n_hypo	The size of the sample used to estimate pg_hypo.
jury_n	Size of the jury (i.e. 6, 8, or 12); default value is 12.
pstrikes	Number of peremptory strikes by prosecution; default value is 0.
dstrikes	Number of peremptory strikes by defendant; default value is 0.

accuracy	Accuracy of parties' peremptory strikes; a number between 0 and 1; default value is .15.
digits	Number of digits to report after decimal places; default value is 3.

Value

Returns a list of jury-level statistics to assess effect of a trial error. Returned list includes statistics for actual jury, hypothetical jury, and the difference between them.

Examples

```
library(sate)
compare.jury.stats(pg_actual=.70, n_actual=400, pg_hypo=.60, n_hypo=450)

compare.jury.stats(pg_actual=.75, n_actual=450, pg_hypo=.65, n_hypo=350,
                   jury_n=6, pstrikes=3, dstrikes=3)
```

---

deliberate *Deliberation function*

Description

The deliberate function returns a jury verdict based on a simulation of deliberation as a modified tug-of-war between two verdict factions. Can be called directly, but is meant to be called many times to generate verdict probabilities based on g_votes and jury_n values.

Usage

```
deliberate(g_votes, jury_n)
```

Arguments

g_votes	Initial number of votes for guilty verdict (same as K value).
jury_n	Size of the jury (i.e. 4, 6, 8, 12, or 16).

Value

Returns "G" (guilty verdict) or "NG" (not guilty verdict).

Examples

```
library(sate)
deliberate(g_votes=10, jury_n=12)

deliberate(g_votes=4, jury_n=6)
```

deliberate.civil *Deliberation function for civil trials (proposed)*

Description

The deliberate function returns a jury verdict based on a simulation of deliberation as a tug-of-war between two verdict factions. The civil version of deliberate does not have presumption in favor of either party. Can be called directly, but is meant to be called many times to generate verdict probabilities based on p_votes and jury_n values.

Usage

```
deliberate.civil(p_votes, jury_n)
```

Arguments

p_votes	Initial number of votes for plaintiff.
jury_n	Size of the jury (i.e. 4, 6, 8, 12, or 16).

Value

Returns "P" (plaintiff verdict) or "D" (defendant verdict).

Examples

```
library(sate)
deliberate.civil(p_votes=8, jury_n=12)

deliberate.civil(p_votes=5, jury_n=6)
```

encode.cloud.respondent.variables *Encodes Cloud Research respondent information in form suitable for calculating sampling weights*

Description

Encodes Cloud research respondent information with names and values suitable for calculating sampling weights. All variables encoded and added to dataset are booleans. The variable respondent_na is TRUE if the respondent's information is "Prefer not to say" or missing on any variable.

Usage

```
encode.cloud.respondent.variables(dataset)
```

Arguments

dataset	Dataset containing Cloud Research respondent demographic information

Value

Returns dataset with encoded variables added: black, ba_or_more, hhincome_over50k, age35plus, woman, hispanic, and respondent_na.

Examples

```
library(sate)

example <- data.frame(Race = sample(x=c("Black or African American", "Other"),
                                    size=10, replace=TRUE),
                      Education = sample(x=c("Bachelor's degree (for example: BA, AB, BS)",
                                             "Other"), size=10, replace=TRUE),
                           Household.Income = sample(x=c("$70,000-$79,999", "Other"),
                                                     size=10, replace=TRUE),
                           Age = sample(x=18:80, size=10, replace=TRUE),
                           Gender = sample(x=c("Woman", "Man", "Prefer not to say"),
                                           size=10, replace=TRUE),
                     Ethnicity = sample(x=c("No, not of Hispanic, Latino, or Spanish origin",
                                             "Other"), size=10, replace=TRUE))
dataset.encoded <- encode.cloud.respondent.variables(dataset=example)
```

---

get_pG_by_k — *Calculates vector of probabilities that jury with n_jurors will return a guilty verdict*

Description

Calculates a vector probabilities that a jury with n_jurors will return a guilty verdict. The vector represents P(G|k) for 0, 1, 2, ... , n_jurors where k is the number of jurors initially in favor of guilty verdict.

Usage

```
get_pG_by_k(n_jurors = 6)
```

Arguments

n_jurors	Size of the jury (i.e. 6, 8, or 12); default value is 6.

Value

Returns a vector of probabilities for guilty verdict of size n_jurors + 1.

Examples

```
library(sate)
get_pG_by_k(10)

get_pG_by_k(n_jurors=12)
```

`graph.effect.defendant` *Plots jury-level differences based on juror-level statistics with effect-on-defendant displayed*

Description

Plots jury-level differences based on juror-level statistics supplied by user. Point estimates supplemented by confidence intervals. Effect-on-defendant also plotted.

Usage

```
graph.effect.defendant(
  pg_actual,
  n_actual,
  pg_hypo,
  n_hypo,
  jury_n = 12,
  pstrikes = 0,
  dstrikes = 0,
  accuracy = 0.15
)
```

Arguments

`pg_actual`	The proportion of jurors who favor a guilty verdict in the actual trial condition (the trial with error).
`n_actual`	The size of the sample used to estimate pg_actual.
`pg_hypo`	The proportion of jurors who favor a guilty verdict in the hypothetical trial condition (the fair trial without error).
`n_hypo`	The size of the sample used to estimate pg_hypo.
`jury_n`	Size of the jury (i.e. 6, 8, or 12); default value is 12.
`pstrikes`	Number of peremptory strikes by prosecution; default value is 0.
`dstrikes`	Number of peremptory strikes by defendant; default value is 0.
`accuracy`	Accuracy of parties' peremptory strikes; a number between 0 and 1; default value is .15.

Value

No return (creates plots)

Examples

```
library(sate)
graph.effect.defendant(pg_actual=.70, n_actual=400, pg_hypo=.60, n_hypo=450)

graph.effect.defendant(pg_actual=.75, n_actual=450, pg_hypo=.65, n_hypo=350,
                       jury_n=6, pstrikes=3, dstrikes=3)
```

`graph.estimate` *Plots probability of a guilty verdict with confidence interval based on juror-level statistics*

Description

Plots probability of guilty verdict with confidence interval based on juror-level statistics supplied by user. Similar to graph.effect.defendant, but plots one condition.

Usage

```
graph.estimate(
  sample_pg,
  sample_n,
  jury_n = 12,
  pstrikes = 0,
  dstrikes = 0,
  accuracy = 0.15
)
```

Arguments

`sample_pg`	The proportion of jurors who favor a guilty verdict in the sample condition
`sample_n`	The size of the sample used to estimate sample_pg_actual
`jury_n`	Size of the jury (i.e. 6, 8, or 12); default value is 12.
`pstrikes`	Number of peremptory strikes by prosecution; default value is 0.
`dstrikes`	Number of peremptory strikes by defendant; default value is 0.
`accuracy`	Accuracy of parties' peremptory strikes; a number between 0 and 1; default value is .15.

Value

No return (creates plot)

Examples

```
library(sate)
graph.estimate(sample_pg=.70, sample_n=400)

graph.estimate(sample_pg=.75, sample_n=450, jury_n=6, pstrikes=3, dstrikes=3)
```

`select.with.strikes` *Generates the distribution of initial votes for guilty verdict on juries*

Description

Calculates and returns probability distribution of initial votes for guilty verdict from 0:jury_n with options for peremptory strikes and strike accuracy. To select jury without strikes, keep pstrikes=0 and dstrikes=0.

Usage

```
select.with.strikes(
  p_g,
  jury_n = 12,
  pstrikes = 0,
  dstrikes = 0,
  accuracy = 0.15
)
```

Arguments

`p_g`	The proportion of jurors in the jury pool who favor a guilty verdict
`jury_n`	Size of the jury (i.e. 6, 8, or 12); default value is 12.
`pstrikes`	Number of peremptory strikes by prosecution; default value is 0.
`dstrikes`	Number of peremptory strikes by defendant; default value is 0.
`accuracy`	Accuracy of parties' peremptory strikes; a number between 0 and 1; default value is .15.

Value

A vector of probabilities for 0:jury_n initial guilty votes

Examples

```
library(sate)
select.with.strikes(p_g=.70, jury_n=6)

select.with.strikes(p_g=.75, jury_n=12, pstrikes=6, dstrikes=10)
```

`sim.as.jury.stats` *Estimates jury-level probability of guilty verdict based on juror-level statistics based on empirical data*

Description

Returns estimate of the probability of guilty verdict based on juror-level statistics supplied by user. Also reports inferential statistics. Results are based on an empirical model with greater uncertainty than as.jury.stats function.

Usage

```
sim.as.jury.stats(
  sample_pg,
  sample_n,
  jury_n = 12,
  pstrikes = 0,
  dstrikes = 0,
  accuracy = 0.15,
  digits = 3,
  nDraws = 10000,
  seed = 12345
)
```

Arguments

`sample_pg`	The proportion of jurors who favor a guilty verdict in the jury pool
`sample_n`	The size of the sample used to estimate sample_pg
`jury_n`	Size of the jury (i.e. 6, 8, or 12); default value is 12.
`pstrikes`	Number of peremptory strikes by prosecution; default value is 0.
`dstrikes`	Number of peremptory strikes by defendant; default value is 0.
`accuracy`	Accuracy of parties' peremptory strikes; a number between 0 and 1; default value is .15.
`digits`	Number of digits to report after decimal places; default value is 3.
`nDraws`	The number of simulations used to generate results. Should be very large number (default = 10000).
`seed`	Set seed for random number generation for replication, default is 12345.

Value

Returns a list of jury-level statistics to assess effect of a trial error.

Examples

```
library(sate)
sim.as.jury.stats(sample_pg=.50, sample_n=830, nDraws=500)

sim.as.jury.stats(sample_pg=10/12, sample_n=295, pstrikes=6, dstrikes=10, nDraws=1000)
```

`sim.compare.jury.stats` *Estimates jury-level differences based on juror-level statistics using simulations based on empirical data*

Description

Calculates jury-level differences based on juror-level statistics supplied by user. Results based on empirical data, inferential statistics produced via simulations.

Usage

```
sim.compare.jury.stats(
  pg_actual,
  n_actual,
  pg_hypo,
  n_hypo,
  jury_n = 12,
  digits = 3,
  pstrikes = 0,
  dstrikes = 0,
  accuracy = 0.15,
  seed = 12345,
  nDraws = 10000
)
```

Arguments

`pg_actual`	The proportion of jurors who favor a guilty verdict in the actual trial condition (the trial with error).
`n_actual`	The size of the sample used to estimate pg_actual.
`pg_hypo`	The proportion of jurors who favor a guilty verdict in the hypothetical trial condition (the fair trial without error).
`n_hypo`	The size of the sample used to estimate pg_hypo.
`jury_n`	Size of the jury (i.e. 6, 8, or 12); default value is 12.
`digits`	Number of digits to report after decimal places; default value is 3.
`pstrikes`	Number of peremptory strikes by prosecution; default value is 0.
`dstrikes`	Number of peremptory strikes by defendant; default value is 0.

accuracy	Accuracy of parties' peremptory strikes; a number between 0 and 1; default value is .15.
seed	Set seed for random number generation for replication, default is 12345.
nDraws	The number of simulations used to generate results. Should be very large number (default = 10000).

Value

Returns a list of jury-level statistics to assess effect of a trial error.

Examples

```
 library(sate)
sim.compare.jury.stats(pg_actual=.70, n_actual=400, pg_hypo=.60, n_hypo=450, nDraws=500)

 sim.compare.jury.stats(pg_actual=.75, n_actual=450, pg_hypo=.65, n_hypo=350,
                   seed=12345, nDraws=1000)
```

---

state.demographic.info

State Demographic Information

Description

A dataset with demographic statistics at state level plus national-level that may be used for calculating sample weights. Includes information related to race, educational attainment, household income, age, gender, and ethnicity.

Usage

```
state.demographic.info
```

Format

A data frame with 52 rows and 8 variables.

state Name of state

StateID Two-letter abbreviation for state. USA for nation.

black Proportion of state population who identify as black (African American), per US Census Bureau.

ba_or_more Proportion of adult (18+) population who have attained a BA degree or more, per US Census Bureau.

hhincome_over50k Proportion of state population with household income of $50,000 or more, per US Census Bureau.

age35plus Proportion of adult (18+) population age 35 or older, per US Census Bureau.

woman Proportion of state population who identify as women, per US Census Bureau.

hispanic Proportion of state population who identify as Hispanic, per US Census Bureau.

Source

U.S. Census Bureau, American Community Survey, 5-Year Estimates.

`target.population.demographics` *Looks up and returns key demographic statistics for target state to be used for calculating sample weights*

Description

Looks up and returns six key demographic statistics for a target state to be used for calculating sample weights. State-level population statistics from U.S. Census Bureau, American Community Survey 5-year estimates. Data from state.demographic.info, a saved datafile in sate package.

Usage

```
target.population.demographics(state)
```

Arguments

`state`	The target state, input as two-letter abbreviation (i.e., "GA" "TX" or "FL"). If no state specified, will use "USA".

Value

A one row data.frame with the following statistics: black, ba_or_more, hhincome_over50k, age35plus, woman, hispanic

Examples

```
library(sate)
target.population.demographics(state="FL")

target.population.demographics()   # will return stats for USA
```

weights_for_population *Calculates survey weights given respondent information and target population demographics*

Description

Calculates survey weights given respondent information and target population demographics. Respondent demographic info must be properly encoded in respondentdata to work with the target.demographics. If respondent demographic info is missing, the respondent's weight will be coded 1. Weight values trimmed so that no weights are greater than 6 or less than .1.

Usage

```
weights_for_population(respondentdata, targetdata)
```

Arguments

respondentdata	Dataset with encoded respondent demographic information (use encode.cloud.respondent.variables to prepare respondentdata) must have a ParticipantId variable.
targetdata	A one row data.frame (or named vector) with the following statistics: black, ba_or_more, hhincome_over50k, age35plus, woman, hispanic (use target.population.demographics to obtain)

Value

Returns respondentdata with raked sampling weights encoded.

Examples

```
library(sate)

example_n <- 100
example <- data.frame(Race = sample(x=c("Black or African American", "Other"),
                                    size=example_n, replace=TRUE),
                      Education = sample(x=c("Bachelor's degree (for example: BA, AB, BS)",
                                             "Other"), size=example_n, replace=TRUE),
                           Household.Income = sample(x=c("$70,000-$79,999", "Other"),
                                                     size=example_n, replace=TRUE),
                           Age = sample(x=18:80, size=example_n, replace=TRUE),
                           Gender = sample(x=c("Woman", "Man", "Prefer not to say"),
                                           size=example_n, replace=TRUE),
                     Ethnicity = sample(x=c("No, not of Hispanic, Latino, or Spanish origin",
                                             "Other"), size=example_n, replace=TRUE),
                           ParticipantId = 1:example_n)
respondents.encoded <- encode.cloud.respondent.variables(dataset=example)

pop.targets <- target.population.demographics(state="FL")
```

```
respondents.weighted <- weights_for_population(respondentdata = respondents.encoded,
                                               targetdata = pop.targets)
```

Index