Package ‘rbw’

September 23, 2021

Type Package

Title Residual Balancing Weights for Marginal Structural Models

Version 0.3.0

Description Residual balancing is a robust method of constructing weights for
marginal structural models, which can be used to estimate (a) the average treatment effect in
a cross-sectional observational study, (b) controlled direct/mediator effects in causal mediation
analysis, and (c) the effects of time-varying treatments in panel data (Zhou and Wodtke 2020
<doi:10.1017/pan.2020.2>). This package provides three functions, rbwATE(), rbwMed(), and rbwPanel(),
that produce residual balancing weights for estimating (a), (b), (c), respectively.

Depends R (>= 3.5.0),
Imports dplyr (>= 0.8.4), stats, rlang (>= 0.4.4)
Suggests knitr, ebal, survey, rmarkdown
License GPL (>= 3)
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1

URL https://github.com/xiangzhou09/rbw

BugReports https://github.com/xiangzhou09/rbw

NeedsCompilation no

Author Xiang Zhou [cre],
        Derick da Silva Baum [aut]

Maintainer Xiang Zhou <xiang_zhou@fas.harvard.edu>

Repository CRAN

Date/Publication 2021-09-22 22:00:02 UTC
R topics documented:

advertisement .......................................................... 2
campaign_long .......................................................... 3
campaign_wide .......................................................... 4
eb2 ............................................................... 5
peace ............................................................. 6
rbwATE .......................................................... 7
rbwMed .......................................................... 9
rbwPanel ......................................................... 10

Index

advertisement 13

Description

A dataset containing 15 variables on the campaign contributions of 16,265 zip codes to the 2004 and 2008 US presidential elections in addition to the demographic characteristics of each area (Urban and Niebler 2014; Fong, Hazlett, and Imai 2018).

Usage

advertisement

Format

A data frame with 16,265 rows and 15 columns:

zip  zip code
treat  the Box-Cox transformed TotAds (Fong, Hazlett, and Imai 2018)
TotAds  the total number of political advertisements aired in the zip code
TotalPop  population size
PercentOver65  percent of the population over 65
Inc  median household income
PercentHispanic  percent Hispanic
PercentBlack  percent black
density  population density (people per sq mile)
per_collegegrads  percent college graduates
CanCommute  a dummy variable indicating whether it is possible to commute to the zip code from a competitive state
StFIPS  state FIPS code
Cont  campaign contributions (in thousands of dollars)
log_TotalPop  log population
log_Inc  log median income
References

campaign_long

Long-format Data on Negative Campaign Advertising in US Senate and Gubernatorial Elections

Description
A dataset containing 19 variables and 565 unit-week records on the campaign of 113 Democratic candidates in US Senate and Gubernatorial Elections from 2000 to 2006 (Blackwell 2013).

Usage
campaign_long

Format
A data frame with 565 rows and 19 columns:

- **demName**: name of the Democratic candidate
- **d.gone.neg**: whether the candidate went negative in a campaign-week, defined as whether more than 10% of the candidate’s political advertising was negative
- **d.gone.neg.l1**: whether the candidate went negative in the previous campaign-week
- **camp.length**: length of the candidate’s campaign (in weeks)
- **deminc**: whether the candidate was an incumbent
- **base.poll**: Democratic share in the baseline polls
- **base.und**: share of undecided voters in the baseline polls
- **office**: type of office in contest. 0: governor; 1: senator
- **demprcnt**: Democratic share of the two-party vote in the election
- **week**: week in the campaign (in the final five weeks preceding the election)
- **year**: year of the election
- **state**: state of the election
- **dem.polls**: Democratic share in the polls
- **dem.polls.l1**: Democratic share in the polls in the previous campaign-week
- **undother**: share of undecided voters in the polls
- **undother.l1**: share of undecided voters in the polls in the previous campaign-week
- **neg.dem**: the proportion of advertisements that were negative in a campaign-week
- **neg.dem.l1**: the proportion of advertisements that were negative in the previous campaign-week
- **id**: candidate id
References

**campaign_wide**

**Wide-format Data on Negative Campaign Advertising in US Senate and Gubernatorial Elections**

**Description**
A dataset containing 32 variables and 113 unit records from Blackwell (2013).

**Usage**
campaign_wide

**Format**
A data frame with 113 rows and 26 columns:
- **demName** name of the Democratic candidate
- **camp.length** length of the candidate’s campaign (in weeks)
- **deminc** whether the candidate was an incumbent.
- **base.poll** Democratic share in the baseline polls
- **base.und** share of undecided voters in the baseline polls
- **office** type of office in contest. 0: governor; 1: senator
- **demprcnt** Democratic share of the two-party vote in the election
- **year** year of the election
- **state** state of the election
- **id** candidate id
- **dem.polls_1** Democratic share in week 1 polls
- **dem.polls_2** Democratic share in week 2 polls
- **dem.polls_3** Democratic share in week 3 polls
- **dem.polls_4** Democratic share in week 4 polls
- **dem.polls_5** Democratic share in week 5 polls
- **d.gone.neg_1** whether the candidate went negative in week 1
- **d.gone.neg_2** whether the candidate went negative in week 2
- **d.gone.neg_3** whether the candidate went negative in week 3
- **d.gone.neg_4** whether the candidate went negative in week 4
- **d.gone.neg_5** whether the candidate went negative in week 5
- **neg.dem_1** the proportion of advertisements that were negative in week 1 polls
neg.dem_2  the proportion of advertisements that were negative in week 2 polls
neg.dem_3  the proportion of advertisements that were negative in week 3 polls
neg.dem_4  the proportion of advertisements that were negative in week 4 polls
neg.dem_5  the proportion of advertisements that were negative in week 5 polls
undother_1  share of undecided voters in week 1 polls
undother_2  share of undecided voters in week 2 polls
undother_3  share of undecided voters in week 3 polls
undother_4  share of undecided voters in week 4 polls
undother_5  share of undecided voters in week 5 polls
cum_neg  the total number of campaign-weeks in which a candidate went negative
ave_neg  the average proportion of advertisements that were negative over the final five weeks of the campaign multiplied by ten

References

---

### eb2

*Function for Generating Minimum Entropy Weights Subject to a Set of Balancing Constraints*

#### Description

`eb2` is an adaptation of `eb` that generates minimum entropy weights subject to a set of balancing constraints. Using the method of Lagrange multipliers, the dual problem is an unconstrained optimization problem that can be solved using Newton’s method. When a full Newton step is excessive, an exact line search is used to find the best step size.

#### Usage

```
   eb2(C, M, Q, Z = rep(0, ncol(C)), max_iter = 200, tol = 1e-04, print_level = 2)
```

#### Arguments

- `C`: A constraint matrix.
- `M`: A vector of moment conditions to be met in the reweighted sample.
- `Q`: A vector of base weights.
- `Z`: A vector of Lagrange multipliers to be initialized.
- `max_iter`: Maximum number of iterations for Newton’s method.
- `tol`: Tolerance parameter used to determine convergence.
- `print_level`: The level of printing:
normal: print whether the algorithm converges or not
2 detailed: print also the maximum absolute value of the deviation between the
moments of the reweighted data and the target moments in each iteration
3 very detailed: print also the step length of the line searcher in iterations where
a full Newton step is excessive.

Value
A list containing the results from the algorithm.

\[ A \text{ vector of normalized minimum entropy weights.} \]
\[ Z \text{ A vector of Lagrange multipliers.} \]
\[ \text{converged} \text{ A logical indicator for convergence.} \]
\[ \text{maxdiff} \text{ A scalar indicating the maximum deviation between the moments of the reweighted} \]
\[ \text{data and the target moments.} \]

Description
A dataset containing 17 variables on the views of 1,273 US adults about their support for war against
countries that were hypothetically developing nuclear weapons. The data include several variables
on the country’s features and respondents’ demographic and attitudinal characteristics (Tomz and

Usage
peace

Format
A data frame with 1,273 rows and 17 columns:

\[ \text{threatc} \text{ number of adverse events respondents considered probable if the US did not engage in war} \]
\[ \text{ally} \text{ a dummy variable indicating whether the country had signed a military alliance with the US} \]
\[ \text{trade} \text{ a dummy variable indicating whether the country had high levels of trade with the US} \]
\[ \text{h1} \text{ an index measuring respondent’s attitude toward militarism} \]
\[ \text{i1} \text{ an index measuring respondent’s attitude toward internationalism} \]
\[ \text{p1} \text{ an index measuring respondent’s identification with the Republican party} \]
\[ \text{e1} \text{ an index measuring respondent’s attitude toward ethnocentrism} \]
\[ \text{r1} \text{ an index measuring respondent’s attitude toward religiosity} \]
\[ \text{male} \text{ a dummy variable indicating whether the respondent is male} \]
\[ \text{white} \text{ a dummy variable indicating whether the respondent is white} \]
**rbwATE**

**Description**

rbwATE is a function that produces residual balancing weights for estimating the average treatment effect (ATE). The weights can be used to fit marginal structural models for the effect of the treatment on the outcome.

**Usage**

rbwATE(
    treatment,  
data,  
    baseline_x,  
    base_weights,  
    max_iter = 200,  
    print_level = 1,  
    tol = 1e-06
)

**age** respondent’s age

**ed4** respondent’s education with categories ranging from high school or less to postgraduate degree

**democ** a dummy variable indicating whether the country was a democracy

**strike** a measure of support for war on a five-point scale

**cost** number of negative consequences anticipated if the US engaged in war

**successc** whether the respondent thought the operation would succeed. 0: less than 50-50 chance of working even in the short run; 1: efficacious only in the short run; 2: successful both in the short and long run

**immoral** a dummy variable indicating whether respondents thought it would be morally wrong to strike the country

**References**


Arguments

treatment  A symbol or character string for the treatment variable.
data  A data frame containing all variables in the model.
baseline_x  An expression for a set of baseline confounders stored in data.
base_weights  (Optional) A vector of base weights (or its name).
max_iter  Maximum number of iterations for Newton’s method.
print_level  The level of printing:
  1  normal: print whether the algorithm converges or not
  2  detailed: print also the maximum absolute value of the deviation between the
          moments of the reweighted data and the target moments in each iteration
  3  very detailed: print also the step length of the line searcher in iterations where
          a full Newton step is excessive.
tol  Tolerance parameter used to determine convergence.

Value

A list containing the results.

weights  A vector of residual balancing weights.
constraints  A matrix of (linearly independent) residual balancing constraints
eb_out  Results from calling eb2 function
call  The matched call.

Examples

# residual balancing weights
rbwATE_fit <- rbwATE(treat, baseline_x = c(log_TotalPop, PercentOver65, log_Inc,
  PercentHispanic, PercentBlack, density, per_collegegrads, CanCommute), data = advertisement)

# attach residual balancing weights to data
advertisement$rbw_ate <- rbwATE_fit$weights

# fit marginal structural model
if(require(survey)){
  rbw_design <- svydesign(ids = ~ 1, weights = ~ rbw_ate, data = advertisement)
  # the outcome model includes the treatment, the square of the treatment,
  # and state-level fixed effects (Fong, Hazlett, and Imai 2018)
  msm_rbwATE <- svyglm(Cont ~ treat + I(treat^2) + factor(StFIPS), design = rbw_design)
  summary(msm_rbwATE)
}

Description

rbwMed is a function that produces residual balancing weights for estimating controlled direct/mediator effects in causal mediation analysis. The weights can be used to fit marginal structural models for the joint effects of the treatment and a mediator.

Usage

rbwMed(
  treatment,
  mediator,
  zmodels,
  data,
  baseline_x,
  interact = FALSE,
  base_weights,
  max_iter = 200,
  print_level = 1,
  tol = 1e-06
)

Arguments

treatment A symbol or character string for the treatment variable.
mediator A symbol or character string for the mediator variable.
zmodels A list of fitted lm or glm objects for post-treatment confounders of the mediator-outcome relationship. If there’s no post-treatment confounder, set it to be NULL.
data A data frame containing all variables in the model.
baseline_x (Optional) An expression for a set of baseline confounders stored in data.
interact A logical variable indicating whether baseline and post-treatment covariates should be balanced against the treatment- mediator interaction term(s).
base_weights (Optional) A vector of base weights (or its name).
max_iter Maximum number of iterations for Newton’s method.
print_level The level of printing:
  1 normal: print whether the algorithm converges or not
  2 detailed: print also the maximum absolute value of the deviation between the moments of the reweighted data and the target moments in each iteration
  3 very detailed: print also the step length of the line searcher in iterations where a full Newton step is excessive.
tol Tolerance parameter used to determine convergence.
Value

A list containing the results.

weights  A vector of residual balancing weights.
constraints  A matrix of (linearly independent) residual balancing constraints
eb_out  Results from calling eb2 function
call  The matched call.

Examples

# models for post-treatment confounders
m1 <- lm(threatc ~ ally + trade + h1 + i1 + p1 + e1 + r1 +
male + white + age + ed4 + democ, data = peace)

m2 <- lm(cost ~ ally + trade + h1 + i1 + p1 + e1 + r1 +
male + white + age + ed4 + democ, data = peace)

m3 <- lm(successc ~ ally + trade + h1 + i1 + p1 + e1 + r1 +
male + white + age + ed4 + democ, data = peace)

# residual balancing weights
rbwMed_fit <- rbwMed(treatment = democ, mediator = immoral,
zmodels = list(m1, m2, m3), interact = TRUE,
baseline_x = c(ally, trade, h1, i1, p1, e1, r1, male, white, age, ed4),
data = peace)

# attach residual balancing weights to data
peace$rbw_cde <- rbwMed_fit$weights

# fit marginal structural model
if(require(survey)){
  rbw_design <- svydesign(ids = ~ 1, weights = ~ rbw_cde, data = peace)
  msm_rbwMed <- svyglm(strike ~ democ * immoral, design = rbw_design)
  summary(msm_rbwMed)
}

rbwPanel  Residual Balancing Weights for Analyzing Time-varying Treatments

Description

rbwPanel is a function that produces residual balancing weights (rbw) for estimating the marginal effects of time-varying treatments. The user supplies a long format data frame (each row being a unit-period) and a list of fitted model objects for time-varying confounders. The residuals of each time-varying covariate $X_t$ are balanced across both current treatment $D_t$ and the regressors of $X_t$. In addition, when $\text{future} > 0$, the residuals are also balanced across future treatments $D_{t+1}, \ldots, D_{t+\text{future}}$.
rbwPanel

Usage

rbwPanel(
  treatment,  # A symbol or character string for the treatment/treatment variable.
  xmodels,    # A list of fitted lm or glm objects for time-varying confounders.
  id,         # A symbol or character string for the unit id variable.
  time,       # A symbol or character string for the time variable. The time variable should be numeric.
  data,       # A data frame containing all variables in the model.
  base_weights,  # (Optional) A vector of base weights (or its name).
  future = 1L,  # An integer indicating the number of future treatments in the balancing conditions. When future > 0, the residualized time-varying covariates are balanced not only with respect to current treatment $D_t$, but also with respect to future treatments $D_{t+1}, \ldots, D_{t+\text{future}}$.
  max_iter = 200,  # Maximum number of iterations for Newton’s method.
  print_level = 1,  # The level of printing:
  tol = 1e-06)      # Tolerance parameter used to determine convergence.
)

Arguments

treatment A symbol or character string for the treatment/treatment variable.
xmodels A list of fitted lm or glm objects for time-varying confounders.
id A symbol or character string for the unit id variable.
time A symbol or character string for the time variable. The time variable should be numeric.
data A data frame containing all variables in the model.
base_weights (Optional) A vector of base weights (or its name).
future An integer indicating the number of future treatments in the balancing conditions. When future > 0, the residualized time-varying covariates are balanced not only with respect to current treatment $D_t$, but also with respect to future treatments $D_{t+1}, \ldots, D_{t+\text{future}}$.
max_iter Maximum number of iterations for Newton’s method.
print_level The level of printing:
  1 normal: print whether the algorithm converges or not
  2 detailed: print also the maximum absolute value of the deviation between the moments of the reweighted data and the target moments in each iteration
  3 very detailed: print also the step length of the line searcher in iterations where a full Newton step is excessive.
tol Tolerance parameter used to determine convergence.

Value

A list containing the results.

weights A data frame containing id and residual balancing weights.
constraints A matrix of (linearly independent) residual balancing constraints
eb_out Results from calling eb2 function
call The matched call.
Examples

# models for time-varying confounders
m1 <- lm(dem.polls ~ (d.gone.neg.l1 + dem.polls.l1 + undother.l1) * factor(week),
data = campaign_long)
m2 <- lm(undother ~ (d.gone.neg.l1 + dem.polls.l1 + undother.l1) * factor(week),
data = campaign_long)

xmodels <- list(m1, m2)

# residual balancing weights
rbwPanel_fit <- rbwPanel(treatment = d.gone.neg, xmodels = xmodels, id = id,
time = week, data = campaign_long)

summary(rbwPanel_fit$weights)

# merge weights into wide-format data
campaign_wide2 <- merge(campaign_wide, rbwPanel_fit$weights, by = "id")

# fit a marginal structural model (adjusting for baseline confounders)
if(require(survey)){
  rbw_design <- svydesign(ids = ~ 1, weights = ~ rbw, data = campaign_wide2)
  msm_rbwPanel <- svyglm(demprcnt ~ cum_neg * deminc + camp.length + factor(year) + office,
                         design = rbw_design)
  summary(msm_rbwPanel)
}
Index

* datasets
  advertisement, 2
  campaign_long, 3
  campaign_wide, 4
  peace, 6
  advertisement, 2
  campaign_long, 3
  campaign_wide, 4
  eb, 5
  eb2, 5, 8, 10, 11
  peace, 6
  rbwATE, 7
  rbwMed, 9
  rbwPanel, 10