Package 'sbw'

December 30, 2019

Type Package

Version 1.0

Date 2019-12-06

Title Stable Balancing Weights for Causal Inference and Estimation with Incomplete Outcome Data

Maintainer Jose R. Zubizarreta <zubizarreta@hcp.med.harvard.edu>

Depends R (>= 3.2), Matrix, quadprog, slam

Imports MASS, spatstat

Enhances gurobi, Rcplex, Rmosek, pogs

License GPL-2 | GPL-3

Description Weights of minimum variance that approximately balance the empirical distribution of the observed covariates.

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

NeedsCompilation no

Author Jose R. Zubizarreta [aut, cre],
Yige Li [aut],
Amine Allouah [ctb],
Noah Greifer [ctb]

Repository CRAN

Date/Publication 2019-12-30 15:20:02 UTC

R topics documented:

estimate *Estimate causal contrasts and population means*

Description

Function for estimating causal contrasts and population means using the output from sbw.

Usage

```
estimate(object, digits = 6, ...)
```

Arguments

object	an object from function sbw.
digits	a scalar with the number of significant digits used to display the estimates. The default is 6.
...	ignored arguments

Examples

```
# Please see the examples in sbw.
```

lalonde *The Lalonde data set*

Description

Data set from the National Supported Work Demonstration used by Lalonde (1986) and Dehejia and Wahba (1999) to evaluate propensity score methods. This data set is publicly available at https://users.nber.org/~rdehejia/data/.nswdata2.html.

Usage

```
data(lalonde)
```

Format

A data frame with 445 observations, corresponding to 185 treated and 260 control subjects, and 10 variables. The treatment assignment indicator is the first variable of the data frame; the next eight columns are the covariates; the last column is the outcome:

treatment the treatment assignment indicator (1 if treated, 0 otherwise)

age a covariate, measured in years

education a covariate, measured in years

black a covariate indicating race (1 if black, 0 otherwise)

hispanic a covariate indicating race (1 if Hispanic, 0 otherwise)

married a covariate indicating marital status (1 if married, 0 otherwise)

nodegree a covariate indicating high school diploma (1 if no degree, 0 otherwise)

re74 a covariate, real earnings in 1974

re75 a covariate, real earnings in 1975

re78 the outcome, real earnings in 1978

Source

https://users.nber.org/~rdehejia/data/.nswdata2.html

References

Dehejia, R., and Wahba, S. (1999), "Causal Effects in Nonexperimental Studies: Reevaluating the Evaluation of Training Programs," *Journal of the American Statistical Association*, 94, 1053-1062.

Lalonde, R. (1986), "Evaluating the Econometric Evaluations of Training Programs," *American Economic Review*, 76, 604-620.

---

sbw *Stable balancing weights for causal contrasts and population means.*

Description

Function for finding stable weights (this is, weights of minimum variance) that approximately balance the empirical distribution of the observed covariates.

Usage

```
sbw(dat, ind = NULL, out = NULL, bal = list(bal_cov, bal_alg = TRUE,
  bal_tol, bal_std = TRUE, bal_gri = c(1e-04, 0.001, 0.002, 0.005, 0.01,
  0.02, 0.05, 0.1), bal_sam = 1000), wei = list(wei_sum = TRUE, wei_pos =
  TRUE), sol = list(sol_nam = "quadprog", sol_dis = FALSE),
  par = list(par_est = "att", par_tar = NULL))
```

Arguments

`dat`	data, a data frame including a treatment or missingness indicator plus covariates; outcomes are optional.
`ind`	treatment or missingness indicator, a string with the name of the binary treatment or missingness, equal to 1 if treated (missing) and 0 otherwise. When `par$par_est = "aux"`, `ind` is omitted.
`out`	outcome, a vector of strings with the names of the outcome variables. The default is `NULL`.

bal balance requirements, a list with requirements for covariate balance with the form list(bal_cov,bal_alg,bal_tol,bal_std,bal_gri,bal_sam), where:

bal_cov: balance covariates, a vector of strings with the names of the covariates in dat to be balanced. In simple applications, the balance covariates in bal_cov will be the column names of dat (of course, without including the treatment or outcome variables) for the original covariates in the data set. The covariates need to be either continuous or binary. Categorical variables need to be transformed into dummy variables. In more complex applications, the covariates in dat can be transformations of the original covariates in order to balance higher order single dimensional moments such as variances and skewness, and multidimensional moments such as correlations. If the transformations of the covariates are indicators of the quantiles of the empirical distribution of a covariate, then balancing all these indicators will tend to balance the entire marginal distribution of the covariate.

bal_alg balance algorithm, a logical that indicates whtether the tuning algorithm in Wang and Zubizarreta (2019) is to be used for automatically selecting the degree of approximate covariates balance. The default is TRUE. See the argument bal_gri below for the candidate values for the degree of approximate covariate balance.

bal_tol: balance tolerances, a scalar or vector of scalars defining the tolerances or maximum differences in means after weighting for the covariates defined in bal_cov. Note that if bal_tol is a vector then its length has to be equal to the length of bal_cov. Otherwise, the first element in the vector will be taken as the balance tolerance for all the constraints in bal_cov.

bal_std: balance tolerances in standard deviations, a logical that indicates whether the tolerances specified in bal_tol are expressed in the original units of the covariates or in standard deviations. The default is TRUE, meaning that the tolerances are expressed in standard deviations.

bal_gri: grid of values for the tuning algorithm bal_alg, a vector of candidate values for the degree of approximate covariate balance. The default is c(0.0001,0.001,0.002,0.005,0.01,0.02,0.05,0.1). The computational time is roughlly proportional to the number of grid values.

bal_sam: number of replicates to be used in bal_alg, an integer specifying the number of bootstrap sample replicates to be used to select the degree of approximate covariate balance. See Wang and Zubizarreta (2019) for details. The default is 1000.

wei weighting constraints, a list with all the weighting constraints with the form list(wei_sum,wei_pos), where:

wei_sum: sum of weights, a logical variable indicating whether the weights are constrained to sum up to one, or their sum is unconstrained. The default is TRUE for the sum of weights equal to one. Note that if wei_sum = TRUE, then wei_pos = TRUE.

wei_pos: positive or zero (non-negative) weights, a logical variable indicating whether the weights are constrained to be non-negative, or they are unconstrained. The default is TRUE for non-negative weights. Again, note that if wei_sum = TRUE, then wei_pos = TRUE.

`sol` solver, a list specifying the solver option with the form `list(sol_nam,sol_dis,sol_pog)` where:

`sol_nam`: solver name, a string equals to one of "cplex", "gurobi", "mosek", "pogs", "quadprog". CPLEX, Gurobi and MOSEK are commercial solvers, but free for academic users. POGS and QUADPROG are free for all. In our experience, POGS is the fastest solver option and able to handle larger datasets, but it can be difficult to install for non-Mac users and more difficult to callibrate. MOSEK is more stable than POGS and fast. The default option is solver = "quadprog".

`sol_dis`: solver display, a logical variable indicating whether the output is to be displayed or not. The default is FALSE. This option is specific to "cplex", "gurobi", "mosek" and "pogs".

`sol_pog`: solver options specific to "pogs", with the following default parameters:

`sol_pog = list(sol_pog_max_iter = 100000,sol_pog_rel_tol = 1e-4, sol_pog_abs_tol = 1e-4,sol_pog_gap_stp = TRUE,sol_pog_adp_rho = TRUE)`.

See the POGS manual for details.

`par` parameter of interest, a list describing the parameter of interest or estimand with the form `list(par_est,par_tar)`, where

`par_est` estimand, a string. For causal inference, a string equals to one of: "att" (Average Treatment effect among the Treated), "atc" (Average Treatment effect among the Controls), "ate" (Average Treatment Effect), "cate" (Conditional Average Treatment Effect). For estimation with incomplete outcome data, a string equal to: "pop" (Population Means) or "aux" (Specified Targets by users). The default is "att".

`par_tar` target, a string or a vector that specifies the targeted population for inference in terms of the observed covariates if `par_est` = "cate", "pop" or "aux" with the form of determine statements. Please see the examples. It also accepts a numeric vector as specified balance targets for `bal$bal_cov`.

Value

A list with the following elements:

`ind`, an inputted argument

`out`, an inputted argument

`bal`, an inputted argument

`wei`, an inputted argument

`sol`, an inputted argument

`par`, an inputted argument

`obj_total`, value/values of the objective function/functions at the optimum;

`eff_size`, effective sample size/sizes for the weighted group/groups;

`time`, time elapsed to find the optimal solution;

`status`, status of the solution. If the optimal weights are found, `status = optimal` ; otherwise, the solution may be not optimal or not exist, in which case an error will be returned with details specific to the solver used. For the solver "quadprog", the status code is missing, therefore, `status = NA` ;

`dat_weights`, data frame with the optimal weights named by weights;

`dual_table`, dual variables or shadow prices of the covariate balancing constraints;

`target`, details of the balance targets, which are saved for evaluation uses.

Source

http://foges.github.io/pogs/stp/r

References

Chattopadhyay, A., Hase, C. H., and Zubizarreta, J. R. (2019), "Balancing Versus Modeling Approaches to Weighting in Practice," submitted.

Kang, J. D. Y., and Schafer, J. L. (2007), "Demistifying Double Robustness: A Comparison of Alternative Strategies for Estimating a Population Mean from Incomplete Data," Statistical Science, 22, 523-539.

Wang, Y., and Zubizarreta, J. R. (2019), "Minimal Dispersion Approximately Balancing Weights: Asymtotic Properties and Practical Considerations," Biometrika, in press.

Zubizarreta, J. R. (2015), "Stable Weights that Balance Covariates for Estimation with Incomplete Outcome Data," Journal of the American Statistical Association, 110, 910-922.

Examples

```
# Simulate data
kangschafer = function(n_obs) {
 # Z are the true covariates
 # t is the indicator for the respondents (treated)
 # y is the outcome
 # X are the observed covariates
 # Returns Z, t y and X sorted in decreasing order by t
 Z = MASS::mvrnorm(n_obs, mu=rep(0, 4), Sigma=diag(4))
 p = 1/(1+exp(Z[, 1]-.5*Z[, 2]+.25*Z[, 3]+.1*Z[, 4]))
 t = rbinom(n_obs, 1, p)
 Zt = cbind(Z, p, t)
 Zt = Zt[order(t), ]
 Z = Zt[, 1:4]
 p = Zt[, 5]
 t = Zt[, 6]
 y = 210+27.4*Z[, 1]+13.7*Z[, 2]+13.7*Z[, 3]+13.7*Z[, 4]+rnorm(n_obs)
 X = cbind(exp(Z[, 1]/2), (Z[, 2]/(1+exp(Z[, 1])))+10, (Z[, 1]*Z[, 3]/
25+.6)^3, (Z[, 2]+Z[, 4]+20)^2)
 return(list(Z=Z, p=p, t=t, y=y, X=X))
}
set.seed(1234)
n_obs = 200
aux = kangschafer(n_obs)
Z = aux$Z
p = aux$p
t = aux$t
y = aux$y
X = aux$X
```

```
# Generate data frame
t_ind = t
bal_cov = X
data_frame = as.data.frame(cbind(t_ind, bal_cov, y))
names(data_frame) = c("t_ind", "X1", "X2", "X3", "X4", "Y")

# Define treatment indicator and
t_ind = "t_ind"
# moment covariates
bal = list()
bal$bal_cov = c("X1", "X2", "X3", "X4")

# Set tolerances
bal$bal_tol = 0.02
bal$bal_std = TRUE

# Solve for the Average Treatment effect among the Treated, ATT (default)
bal$bal_alg = FALSE
sbwatt.object = sbw(dat = data_frame, ind = t_ind, out = "Y", bal = bal)

# # Solve for a Conditional Average Treatment Effect, CATE
# sbwcate.object = sbw(dat = data_frame, ind = t_ind, out = "Y", bal = bal,
# sol = list(sol_nam = "quadprog"), par = list(par_est = "cate", par_tar = "X1 > 1 & X3 <= 0.22"))

# # Solve for the population mean, POP
# tar = colMeans(bal_cov)
# names(tar) = bal$bal_cov
# sbwpop.object = sbw(dat = data_frame, ind = t_ind, out = "Y", bal = bal,
# sol = list(sol_nam = "quadprog"), par = list(par_est = "pop"))

# # Solve for a target population mean, AUX
# sbwaux.object = sbw(dat = data_frame, bal = bal,
# sol = list(sol_nam = "quadprog"), par = list(par_est = "aux", par_tar = tar*1.05))

# # Solve for the ATT using the tuning algorithm
# bal$bal_alg = TRUE
# bal$bal_sam = 1000
# sbwatttun.object = sbw(dat = data_frame, ind = t_ind, out = "Y", bal = bal,
# sol = list(sol_nam = "quadprog"), par = list(par_est = "att", par_tar = NULL))

# Check
summarize(sbwatt.object)
# summarize(sbwcate.object)
# summarize(sbwpop.object)
# summarize(sbwaux.object)
# summarize(sbwatttun.object)

# Estimate
estimate(sbwatt.object)
# estimate(sbwcate.object)
# estimate(sbwpop.object)
# estimate(sbwatttun.object)
```

```
# Visualize
visualize(sbwatt.object)
# visualize(sbwcate.object)
# visualize(sbwpop.object)
# visualize(sbwaux.object)
# visualize(sbwatttun.object)
```

---

summarize *Visualize output from* sbw

Description

Function for summarizing the output from sbw.

Usage

```
summarize(object, digits = 6, ...)
```

Arguments

object	an object from the class sbwcau or sbwpop obtained after using sbw.
digits	The number of significant digits that will be displayed. The default is 6.
...	ignored arguments

Examples

```
# Please see the examples in function sbw.
```

---

visualize *Visualize output from* sbw

Description

Function for visualizing the output from sbw.

Usage

```
visualize(object, plot_cov, ...)
```

Arguments

object	an object from function sbw.
plot_cov	names of covariates for which balance is to be displayed. If NULL, all of the covariates will be displayed.
...	ignored

Examples

```
# Please see the examples in \code{sbw}.
```

Index