Package ‘CBPS’

January 18, 2022

Version 0.23
Date 2022-01-18
Title Covariate Balancing Propensity Score
Depends R (>= 3.4), MASS, MatchIt, nnet, numDeriv, glmnet

Description
Implements the covariate balancing propensity score (CBPS) proposed by Imai and Ratkovic (2014) <DOI:10.1111/rssb.12027>. The propensity score is estimated such that it maximizes the resulting covariate balance as well as the prediction of treatment assignment. The method, therefore, avoids an iteration between model fitting and balance checking. The package also implements optimal CBPS from Fan et al. (in-press) <DOI:10.1080/07350015.2021.2002159>, several extensions of the CBPS beyond the cross-sectional, binary treatment setting. They include the CBPS for longitudinal settings so that it can be used in conjunction with marginal structural models from Imai and Ratkovic (2015) <DOI:10.1080/01621459.2014.956872>, treatments with three- and four-valued treatment variables, continuous-valued treatments from Fong, Hazlett, and Imai (2018) <DOI:10.1214/17-AOAS1101>, propensity score estimation with a large number of covariates from Ning, Peng, and Imai (2020) <DOI:10.1093/biomet/asaa020>, and the situation with multiple distinct binary treatments administered simultaneously. In the future it will be extended to other settings including the generalization of experimental and instrumental variable estimates.

LazyLoad yes
LazyData yes
License GPL (>= 2)
NeedsCompilation no
Repository CRAN
RoxygenNote 7.1.2
Suggests testthat
Author Christian Fong [aut, cre],
Marc Ratkovic [aut],
Kosuke Imai [aut],
Chad Hazlett [ctb],
AsyVar

Asymptotic Variance and Confidence Interval Estimation of the ATE

Description

AsyVar estimates the asymptotic variance of the ATE obtained with the CBPS or oCBPS method. It also returns the finite variance estimate, the finite standard error, and a CI for the ATE.
Usage

AsyVar(
  Y,
  Y_1_hat = NULL,
  Y_0_hat = NULL,
  CBPS_obj,
  method = "CBPS",
  X = NULL,
  TL = NULL,
  pi = NULL,
  mu = NULL,
  CI = 0.95
)

Arguments

Y
  The vector of actual outcome values (observations).

Y_1_hat
  The vector of estimated outcomes according to the treatment model. (AsyVar automatically sets the treatment model as a linear regression model and it is fitted within the function.) If CBPS_obj is specified, or if X and TL are specified, this is unnecessary.

Y_0_hat
  The vector of estimated outcomes according to the control model. (AsyVar automatically sets the control model as a linear regression model and it is fitted within the function.) If CBPS_obj is specified, or if X and TL are specified, this is unnecessary.

CBPS_obj
  An object obtained with the CBPS function. If this object is not specified, then X, TL, pi, and mu must all be specified instead.

method
  The specific method to be considered. Either "CBPS" or "oCBPS" must be selected.

X
  The matrix of covariates with the rows corresponding to the observations and the columns corresponding to the variables. The left most column must be a column of 1’s for the intercept. (X is not necessary if CBPS_obj is specified.)

TL
  The vector of treatment labels. More specifically, the label is 1 if it is in the treatment group and 0 if it is in the control group. (TL is not necessary if CBPS_obj is specified.)

pi
  The vector of estimated propensity scores. (pi is not necessary if CBPS_obj is specified.)

mu
  The estimated average treatment effect obtained with either the CBPS or oCBPS method. (mu is not necessary if CBPS_obj is specified.)

CI
  The specified confidence level (between 0 and 1) for calculating the confidence interval for the average treatment effect. Default value is 0.95.

Value

mu.hat
  The estimated average treatment effect, hatµ.
The estimated asymptotic variance of $\sqrt{n} \cdot \hat{\mu}$ obtained with the CBPS or oCBPS method.

The estimated variance of $\hat{\mu}$ obtained with the CBPS or oCBPS method.

The standard error of $\hat{\mu}$ obtained with the CBPS or oCBPS method.

The confidence interval of $\hat{\mu}$ obtained with the CBPS or oCBPS method with the confidence level specified in the input argument.

Author(s)

Inbeom Lee

References


Examples

#GENERATING THE DATA
n=300

#Initialize the X matrix.
X_v1 <- rnorm(n,3,sqrt(2))
X_v2 <- rnorm(n,0,1)
X_v3 <- rnorm(n,0,1)
X_v4 <- rnorm(n,0,1)
X_mat <- as.matrix(cbind(rep(1,n), X_v1, X_v2, X_v3, X_v4))

#Initialize the Y_1 and Y_0 vector using the treatment model and the control model.
Y_1 <- X_mat %*% matrix(c(200, 27.4, 13.7, 13.7, 13.7), 5, 1) + rnorm(n)
Y_0 <- X_mat %*% matrix(c(200, 0 , 13.7, 13.7, 13.7), 5, 1) + rnorm(n)

#True Propensity Score calculation.
pre_prop <- X_mat[,2:5] %*% matrix(c(0, 0.5, -0.25, -0.1), 4, 1)
propensity_true <- (exp(pre_prop))/(1+(exp(pre_prop))

#Generate T_vec, the treatment vector, with the true propensity scores.
T_vec <- rbinom(n, size=1, prob=propensity_true)

#Now generate the actual outcome Y_outcome (accounting for treatment/control groups).
Y_outcome <- Y_1*T_vec + Y_0*(1-T_vec)

#Use oCBPS.
ocbps.fit <- CBPS(T_vec ~ X_mat, ATT=0, baseline.formula = ~X_mat[,c(1,3:5)],
                diff.formula = ~X_mat[,2])

#Use the AsyVar function to get the asymptotic variance of the estimated average treatment effect and its confidence interval when using oCBPS.
AsyVar(Y=Y_outcome, CBPS_obj=ocbps.fit, method="oCBPS", CI=0.95)
balance

# Use CBPS.
cbps.fit <- CBPS(T_vec ~ X_mat, ATT=0)

# Use the AsyVar function to get the asymptotic variance of the
# estimated average treatment effect and its confidence interval when using CBPS.
AsyVar(Y=Y_outcome, CBPS_obj=cbps.fit, method="CBPS", CI=0.95)

---

### balance  
*Optimal Covariate Balance*

**Description**

Returns the mean and standardized mean associated with each treatment group, before and after weighting. To access more comprehensive diagnostics for assessing covariate balance, consider using Noah Greifer's cobalt package.

**Usage**

balance(object, ...)

**Arguments**

- **object**: A CBPS, npCBPS, or CBMSM object.
- **...**: Additional arguments to be passed to balance.

**Details**

For binary and multi-valued treatments as well as marginal structural models, each of the matrices' rows are the covariates and whose columns are the weighted mean, and standardized mean associated with each treatment group. The standardized mean is the weighted mean divided by the standard deviation of the covariate for the whole population. For continuous treatments, returns the absolute Pearson correlation between the treatment and each covariate.

##### @aliases balance balance.npCBPS balance.CBPS balance.CBMSM

**Value**

Returns a list of two matrices, "original" (before weighting) and "balanced" (after weighting).

**Author(s)**

Christian Fong, Marc Ratkovic, and Kosuke Imai.
Examples

```r
### Example: Assess Covariate Balance
###
data(LaLonde)
## Estimate CBPS
fit <- CBPS(treat ~ age + educ + re75 + re74 + I(re75==0) + I(re74==0),
data = LaLonde, ATT = TRUE)
balance(fit)
```

#### balance.CBPS

*Calculates the pre- and post-weighting difference in standardized means for covariate within each contrast*

**Description**

Calculates the pre- and post-weighting difference in standardized means for covariate within each contrast

**Usage**

```r
## S3 method for class 'CBPS'
balance(object, ...)
```

**Arguments**

- **object**: A CBPS, npCBPS, or CBMSM object.
- **...**: Additional arguments to be passed to balance.

#### balance.CBPSContinuous

*Calculates the pre- and post-weighting correlations between each covariate and the T*

**Description**

Calculates the pre- and post-weighting correlations between each covariate and the T

**Usage**

```r
## S3 method for class 'CBPSContinuous'
balance(object, ...)
```
**balance.npCBPS**

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>A CBPS, npCBPS, or CBMSM object.</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments to be passed to balance.</td>
</tr>
</tbody>
</table>

**Description**

Calls the appropriate balance function based on the number of treatments.

**Usage**

```r
## S3 method for class 'npCBPS'
balance(object, ...)```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>A CBPS, npCBPS, or CBMSM object.</td>
</tr>
<tr>
<td>...</td>
<td>Other parameters to be passed.</td>
</tr>
</tbody>
</table>

**Blackwell**

**Blackwell Data for Covariate Balancing Propensity Score**

**Description**

This data set gives the outcomes as well as treatment assignments and covariates for the example from Blackwell (2013).

**Format**

A data frame consisting of 13 columns (including treatment assignment, time, and identifier vectors) and 570 observations.

**Source**

d.gone.neg is the treatment. d.gone.neg.l1, d.gone.neg.l2, and d.gone.neg.l3 are lagged treatment variables. camp.length, deminc, base.poll, base.und, and office covariates. year is the year of the particular race, and time goes from the first measurement (time = 1) to the election (time = 5). demName is the identifier, and demprcnt is the outcome.

**References**

Covariate Balancing Propensity Score for Instrumental Variable Estimates (CBIV)

Description

CBIV estimates propensity scores for compliance status in an instrumental variables setup such that both covariate balance and prediction of treatment assignment are maximized. The method, therefore, avoids an iterative process between model fitting and balance checking and implements both simultaneously.

Usage

CBIV(
    Tr,
    Z,
    X,
    iterations = 1000,
    method = "over",
    twostep = TRUE,
    twosided = TRUE,
    ...
)

Arguments

Tr  A binary treatment variable.
Z  A binary encouragement variable.
X  A pre-treatment covariate matrix.
iterations  An optional parameter for the maximum number of iterations for the optimization. Default is 1000.
method  Choose "over" to fit an over-identified model that combines the propensity score and covariate balancing conditions; choose "exact" to fit a model that only contains the covariate balancing conditions. Our simulations suggest that "over" dramatically outperforms "exact."

twostep  Default is TRUE for a two-step GMM estimator, which will run substantially faster than continuous-updating. Set to FALSE to use the continuous-updating GMM estimator.

twosided  Default is TRUE, which allows for two-sided noncompliance with both always-takers and never-takers. Set to FALSE for one-sided noncompliance, which allows only for never-takers.

...  Other parameters to be passed through to optim().
Details

Fits covariate balancing propensity scores for generalizing local average treatment effect estimates obtained from instrumental variables analysis.

Value

- **coefficients**: A named matrix of coefficients, where the first column gives the complier coefficients and the second column gives the always-taker coefficients.
- **fitted.values**: The fitted N x 3 compliance score matrix. The first column gives the estimated probability of being a complier, the second column gives the estimated probability of being an always-taker, and the third column gives the estimated probability of being a never-taker.
- **weights**: The optimal weights: the reciprocal of the probability of being a complier.
- **deviance**: Minus twice the log-likelihood of the CBIV fit.
- **converged**: Convergence value. Returned from the call to `optim()`.
- **J**: The J-statistic at convergence
- **df**: The number of linearly independent covariates.
- **bal**: The covariate balance associated with the optimal weights, calculated as the GMM loss of the covariate balance conditions.

Author(s)

Christian Fong

References


Examples

```r
### Example: propensity score matching
### (Need to fix when we have an actual example).

#Load the LaLonde data
data(LaLonde)
## Estimate CBPS
fit <- CBPS(treat ~ age + educ + re75 + re74 + I(re75==0) + I(re74==0),
            data = LaLonde, ATT = TRUE)
summary(fit)
```
CBMSM

*Covariate Balancing Propensity Score (CBPS) for Marginal Structural Models*

**Description**

CBMSM estimates propensity scores such that both covariate balance and prediction of treatment assignment are maximized. With longitudinal data, the method returns marginal structural model weights that can be entered directly into a linear model. The method also handles multiple binary treatments administered concurrently.

**Usage**

```
CBMSM(
  formula,           
  id,                
  time,              
  data,              
  type = "MSM",     
  twostep = TRUE,    
  msm.variance = "approx", 
  time.vary = FALSE, 
  init = "opt",     
  ...               
)
```

**Arguments**

- **formula**: A formula of the form `treat ~ X`. The same covariates are used in each time period. At default values, a single set of coefficients is estimated across all time periods. To allow a different set of coefficients for each time period, set `time.vary = TRUE`. Data should be sorted by time.
- **id**: A vector which identifies the unit associated with each row of `treat` and `X`.
- **time**: A vector which identifies the time period associated with each row of `treat` and `X`. All data should be sorted by time.
- **data**: An optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `CBMSM` is called. Data should be sorted by time.
- **type**: "MSM" for a marginal structural model, with multiple time periods or "Multi-Bin" for multiple binary treatments at the same time period.
- **twostep**: Set to `TRUE` to use a two-step estimator, which will run substantially faster than continuous-updating. Default is `FALSE`, which uses the continuous-updating estimator described by Imai and Ratkovic (2014).
- **msm.variance**: Default is `FALSE`, which uses the low-rank approximation of the variance described in Imai and Ratkovic (2014). Set to `TRUE` to use the full variance matrix.
time.vary Default is FALSE, which uses the same coefficients across time period. Set to TRUE to fit one set per time period.

init Default is "opt", which uses CBPS and logistic regression starting values, and chooses the one that achieves the best balance. Other options are "glm" and "CBPS"

... Other parameters to be passed through to optim()

Details

Fits covariate balancing propensity scores for marginal structural models.

### @aliases CBMSM CBMSM.fit

Value

weights The optimal weights.

fitted.values The fitted propensity score for each observation.

y The treatment vector used.

x The covariate matrix.

id The vector id used in CBMSM.fit.

time The vector time used in CBMSM.fit.

model The model frame.

call The matched call.

formula The formula supplied.

data The data argument.

treat.hist A matrix of the treatment history, with each observation in rows and time in columns.

treat.cum A vector of the cumulative treatment history, by individual.

Author(s)

Marc Ratkovic, Christian Fong, and Kosuke Imai; The CBMSM function is based on the code for version 2.15.0 of the glm function implemented in the stats package, originally written by Simon Davies. This documentation is likewise modeled on the documentation for glm and borrows its language where the arguments and values are the same.

References


See Also

plot.CBMSM

Examples

```r
# Load Blackwell data
data(Blackwell)

## Quickly fit a short model to test
form0 <- "d.gone.neg ~ d.gone.neg.l1 + camp.length"
fit0 <- CBMSM(formula = form0, time=Blackwell$time, id=Blackwell$demName, 
data=Blackwell, type="MSM", iterations = NULL, twostep = TRUE, 
msm.variance = "approx", time.vary = FALSE)

## Not run:
## Fitting the models in Imai and Ratkovic (2014)
## Warning: may take a few minutes; setting time.vary to FALSE
## Results in a quicker fit but with poorer balance
## Usually, it is best to use time.vary TRUE
form1 <- "d.gone.neg ~ d.gone.neg.l1 + d.gone.neg.l2 + d.neg.frac.l3 + 
camp.length + camp.length + deminc + base.poll + year.2002 + 
year.2004 + year.2006 + base.und + office"

## Note that init="glm" gives the published results but the default is now init="opt"
fit1 <- CBMSM(formula = form1, time=Blackwell$time, id=Blackwell$demName, 
data=Blackwell, type="MSM", iterations = NULL, twostep = TRUE, 
msm.variance = "full", time.vary = TRUE, init="glm")

fit2 <- CBMSM(formula = form1, time=Blackwell$time, id=Blackwell$demName, 
data=Blackwell, type="MSM", iterations = NULL, twostep = TRUE, 
msm.variance = "approx", time.vary = TRUE, init="glm")

## Assessing balance
bal1 <- balance.CBMSM(fit1)
bal2 <- balance.CBMSM(fit2)

## Effect estimation: Replicating Effect Estimates in 
## Table 3 of Imai and Ratkovic (2014)
lm1 <- lm(demprcnt[time==1] ~ fit1$treat.hist, data=Blackwell, 
weights=fit1$glm.weights)
lm2 <- lm(demprcnt[time==1] ~ fit1$treat.hist, data=Blackwell, 
weights=fit1$weights)
lm3 <- lm(demprcnt[time==1] ~ fit1$treat.hist, data=Blackwell, 
weights=fit2$weights)
lm4 <- lm(demprcnt[time==1] ~ fit1$treat.cum, data=Blackwell, 
weights=fit2$weights)
```

weights=fit1$glm.weights)

lm5<-lm(demprcnt[time==1]-fit1$treat.cum,data=Blackwell, weights=fit1$weights)

lm6<-lm(demprcnt[time==1]-fit1$treat.cum,data=Blackwell, weights=fit2$weights)

### Example: Multiple Binary Treatments Administered at the Same Time

n<-200
k<-4
set.seed(1040)
X1<-cbind(1,matrix(rnorm(n*k),ncol=k))

betas.1<-betas.2<-betas.3<-c(2,4,4,-4,3)/5
probs.1<-probs.2<-probs.3<-(-1+exp(-X1 %*% betas.1))^(-1)

treat.1<-rbinom(n=length(probs.1),size=1,probs.1)
treat.2<-rbinom(n=length(probs.2),size=1,probs.2)
treat.3<-rbinom(n=length(probs.3),size=1,probs.3)
treat<-c(treat.1,treat.2,treat.3)
X<-rbind(X1,X1,X1)
time<-c(rep(1,nrow(X1)),rep(2,nrow(X1)),rep(3,nrow(X1)))

id<-c(rep(1:nrow(X1),3))
y<-cbind(treat.1,treat.2,treat.3) %*% c(2,2,2) +
X1 %*% c(-2,8,7,6,2) + rnorm(n,sd=5)

multibin1<-CBMSM(treat~X,id=id,time=time,type="MultiBin",twostep=TRUE)

summary(lm(y~-1+treat.1+treat.2+treat.3+X1, weights=multibin1$w))

## End(Not run)

---

**Description**

CBMSM.fit

**Usage**

CBMSM.fit(
  treat,
  X,
  id,
  time,
  MultiBin.fit,
  twostep,
  msm.variance,
)
time.vary,
ininit,
...
)

Arguments

treat A vector of treatment assignments. For N observations over T time periods, the length of treat should be N*T.

X A covariate matrix. For N observations over T time periods, X should have N*T rows.

id A vector which identifies the unit associated with each row of treat and X.

time A vector which identifies the time period associated with each row of treat and X.

MultiBin.fit A parameter for whether the multiple binary treatments occur concurrently (FALSE) or over consecutive time periods (TRUE) as in a marginal structural model. Setting type = "MultiBin" when calling CBMSM will set MultiBin.fit to TRUE when CBMSM.fit is called.

twostep Set to TRUE to use a two-step estimator, which will run substantially faster than continuous-updating. Default is FALSE, which uses the continuous-updating estimator described by Imai and Ratkovic (2014).

msm.variance Default is FALSE, which uses the low-rank approximation of the variance described in Imai and Ratkovic (2014). Set to TRUE to use the full variance matrix.

time.vary Default is FALSE, which uses the same coefficients across time period. Set to TRUE to fit one set per time period.

init Default is "opt", which uses CBPS and logistic regression starting values, and chooses the one that achieves the best balance. Other options are "glm" and "CBPS"

 Other parameters to be passed through to optim()
Usage

```r
CBPS(
    formula,
    data,
    na.action,
    ATT = 1,
    iterations = 1000,
    standardize = TRUE,
    method = "over",
    twostep = TRUE,
    sample.weights = NULL,
    baseline.formula = NULL,
    diff.formula = NULL,
    ...
)
```

Arguments

- **formula**: An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted.

- **data**: An optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in data, the variables are taken from `environment(formula)`, typically the environment from which `CBPS` is called.

- **na.action**: A function which indicates what should happen when the data contain NAs. The default is set by the `na.action` setting of options, and is `na.fail` if that is unset.

- **ATT**: Default is 1, which finds the average treatment effect on the treated interpreting the second level of the treatment factor as the treatment. Set to 2 to find the ATT interpreting the first level of the treatment factor as the treatment. Set to 0 to find the average treatment effect. For non-binary treatments, only the ATE is available.

- **iterations**: An optional parameter for the maximum number of iterations for the optimization. Default is 1000.

- **standardize**: Default is TRUE, which normalizes weights to sum to 1 within each treatment group. For continuous treatments, normalizes weights to sum up to 1 for the entire sample. Set to FALSE to return Horvitz-Thompson weights.

- **method**: Choose "over" to fit an over-identified model that combines the propensity score and covariate balancing conditions; choose "exact" to fit a model that only contains the covariate balancing conditions.

- **twostep**: Default is TRUE for a two-step estimator, which will run substantially faster than continuous-updating. Set to FALSE to use the continuous-updating estimator described by Imai and Ratkovic (2014).

- **sample.weights**: Survey sampling weights for the observations, if applicable. When left NULL, defaults to a sampling weight of 1 for each observation.
baseline.formula
Used only to fit iCBPS (see Fan et al). Currently only works with binary treatments. A formula specifying the balancing covariates in the baseline outcome model, i.e., \( E(Y(0)|X) \).

diff.formula
Used only to fit iCBPS (see Fan et al). Currently only works with binary treatments. A formula specifying the balancing covariates in the difference between the treatment and baseline outcome model, i.e., \( E(Y(1)-Y(0)|X) \).

Value

- **fitted.values**: The fitted propensity score
- **linear.predictor**: \( X \ast \beta \)
- **deviance**: Minus twice the log-likelihood of the CBPS fit
- **weights**: The optimal weights. Let \( \pi_i = f(T_i|X_i) \). For binary ATE, these are given by \( \frac{T_i}{\pi_i} + \frac{(1-T_i)}{1-\pi_i} \). For binary ATT, these are given by \( \frac{\pi_i}{\pi_i} \ast \frac{T_i}{1-\pi_i} \). For multi-valued treatments, these are given by \( \sum_{j=0}^{J-1} T_{i,j}/\pi_{i,j} \). For continuous treatments, these are given by \( \frac{f(T_i)}{\frac{f(T_i)}{f(T_i|X_i)}} \). These expressions for weights are all before standardization (i.e. with standardize=FALSE). Standardization will make weights sum to 1 within each treatment group. For continuous treatment, standardization will make all weights sum to 1. If sampling weights are used, the weight for each observation is multiplied by the survey sampling weight.

- **y**: The treatment vector used
- **x**: The covariate matrix
- **model**: The model frame
- **converged**: Convergence value. Returned from the call to `optim()`.
- **call**: The matched call
- **formula**: The formula supplied
- **data**: The data argument
- **coefficients**: A named vector of coefficients
- **sigmasq**: The sigma-squared value, for continuous treatments only
- **J**: The J-statistic at convergence
- **mle.J**: The J-statistic for the parameters from maximum likelihood estimation
- **var**: The covariance matrix for the coefficients.
- **Ttilde**: For internal use only.
- **Xtilde**: For internal use only.
- **beta.tilde**: For internal use only.
- **simgasq.tilde**: For internal use only.
Author(s)
Christian Fong, Marc Ratkovic, Kosuke Imai, and Xiaolin Yang; The CBPS function is based on the code for version 2.15.0 of the glm function implemented in the stats package, originally written by Simon Davies. This documentation is likewise modeled on the documentation for glm and borrows its language where the arguments and values are the same.

References

See Also
summary.CBPS

Examples

```r
### Example: propensity score matching

```load the LaLonde data
data(LaLonde)
```Estimate CBPS
```fit <- CBPS(treat ~ age + educ + re75 + re74 +
I(re75==0) + I(re74==0),
data = LaLonde, ATT = TRUE)
```summary(fit)
```# Not run:
# matching via MatchIt: one to one nearest neighbor with replacement
library(MatchIt)
m.out <- matchit(treat ~ fitted(fit), method = "nearest",
data = LaLonde, replace = TRUE)
```### Example: propensity score weighting
```### Simulation from Kang and Shafer (2007).
set.seed(123456)
n <- 500
X <- mvrnorm(n, mu = rep(0, 4), Sigma = diag(4))
prop <- 1 / (1 + exp(X[,1] - 0.5 * X[,2] +
0.25*X[,3] + 0.1 * X[,4]))
treat <- rbinom(n, 1, prop)
```
\[
y \leftarrow 210 + 27.4 \times X_{[1]} + 13.7 \times X_{[2]} + 13.7 \times X_{[3]} + 13.7 \times X_{[4]} + \text{rnorm}(n)
\]

### Estimate CBPS with a misspecified model
\[
X_{\text{mis}} \leftarrow \text{cbind}(\exp(X_{[1]}/2), X_{[2]} \times (1 + \exp(X_{[1]}))^{(-1)} + 10, \\
(X_{[1]} X_{[3]}/25 + .6)^{3}, \ (X_{[2]} + X_{[4]} + 20)^{2})
\]
\[
\text{fit1} \leftarrow \text{CBPS}(\text{treat} \sim X_{\text{mis}}, \text{ATT} = 0)
\]
\[
\text{summary(fit1)}
\]

### Horwitz-Thompson estimate
\[
\text{mean(treat} \times y/\text{fit1}\$\text{fitted.values})
\]

### Inverse propensity score weighting
\[
\text{sum(treat} \times y/\text{fit1}\$\text{fitted.values})/\text{sum(treat}/\text{fit1}\$\text{fitted.values})
\]

\[
\text{rm(list=c("y","X","prop","treat","n","X.mis","fit1"))}
\]

### Example: Continuous Treatment as in Fong, Hazlett, and Imai (2018). See
### https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/AIF4PI
### for a real data example.
\[
\text{set.seed(123456)}
\]
\[
n \leftarrow 1000
\]
\[
X \leftarrow \text{mvnrnorm}(n, \text{mu} = \text{rep(0,2)}, \text{Sigma} = \text{diag}(2))
\]
\[
\beta \leftarrow \text{rnorm(ncol(X)+1, sd = 1)}
\]
\[
\text{treat} \leftarrow \text{cbind}(1, X) \times \beta + \text{rnorm}(n, \text{sd} = 5)
\]
\[
\text{treat.effect} \leftarrow 1
\]
\[
\text{effect.beta} \leftarrow \text{rnorm(ncol(X))}
\]
\[
y \leftarrow \text{rbinom}(n, 1, (1 + \exp(-\text{treat.effect} \times \text{treat} - X) \times \text{effect.beta}))^{(-1)}
\]
\[
\text{fit2} \leftarrow \text{CBPS(}\text{treat} \sim X)
\]
\[
\text{summary(fit2)}
\]
\[
\text{summary(glm(y} \sim \text{treat} \times X, \text{weights} = \text{fit2}\$\text{weights, family = "quasibinomial"))}
\]

\[
\text{rm(list=c("n","X","beta","treat","treat.effect","effect.beta","y","fit2"))}
\]

### Simulation example: Improved CBPS (or iCBPS) from Fan et al
set.seed(123456)
\[
n \leftarrow 500
\]
\[
X \leftarrow \text{mvnrnorm}(n, \text{mu} = \text{rep}(0,4), \text{Sigma} = \text{diag}(4))
\]
\[
\text{prop} \leftarrow 1 / (1 + \exp(X_{[1]} - 0.5 \times X_{[2]} + 0.25 \times X_{[3]} + 0.1 \times X_{[4]}))
\]
\[
\text{treat} \leftarrow \text{rbinom}(n, 1, \text{prop})
\]
\[
y1 \leftarrow 210 + 27.4 \times X_{[1]} + 13.7 \times X_{[2]} + 13.7 \times X_{[3]} + 13.7 \times X_{[4]} + \text{rnorm}(n)
\]
\[
y0 \leftarrow 210 + 13.7 \times X_{[2]} + 13.7 \times X_{[3]} + 13.7 \times X_{[4]} + \text{rnorm}(n)
\]
\[
\text{#Estimate iCBPS with a misspecified model}
\]
\[
X_{\text{mis}} \leftarrow \text{cbind}(\exp(X_{[1]}/2), X_{[2]} \times (1 + \exp(X_{[1]}))^{(-1)} + 10, \\
(X_{[1]} X_{[3]}/25 + .6)^{3}, \ (X_{[2]} + X_{[4]} + 20)^{2})
\]
\[
\text{fit1} \leftarrow \text{CBPS(}\text{treat} \sim X_{\text{mis}}, \text{baseline.formula} = X_{\text{mis}}[1:2], \text{diff.formula} = X_{\text{mis}}[2:4], \text{ATT} = \text{FALSE})
\]
\[
\text{summary(fit1)}
\]
CBPS.fit determines the proper routine (what kind of treatment) and calls the appropriate function. It also pre- and post-processes the data.

Description

CBPS.fit determines the proper routine (what kind of treatment) and calls the appropriate function. It also pre- and post-processes the data.

Usage

```r
CBPS.fit(
  treat,
  X,
  baselineX,
  diffX,
  ATT,
  method,
  iterations,
  standardize,
  twostep,
  sample.weights = sample.weights,
  ...
)
```

Arguments

treat: A vector of treatment assignments. Binary or multi-valued treatments should be factors. Continuous treatments should be numeric.

X: A covariate matrix.

baselineX: Similar to baseline.formula, but in matrix form.

diffX: Similar to diff.formula, but in matrix form.

ATT: Default is 1, which finds the average treatment effect on the treated interpreting the second level of the treatment factor as the treatment. Set to 2 to find the ATT interpreting the first level of the treatment factor as the treatment. Set to 0 to find the average treatment effect. For non-binary treatments, only the ATE is available.

method: Choose "over" to fit an over-identified model that combines the propensity score and covariate balancing conditions; choose "exact" to fit a model that only contains the covariate balancing conditions.

iterations: An optional parameter for the maximum number of iterations for the optimization. Default is 1000.
standardize Default is TRUE, which normalizes weights to sum to 1 within each treatment group. For continuous treatments, normalizes weights to sum up to 1 for the entire sample. Set to FALSE to return Horvitz-Thompson weights.

twostep Default is TRUE for a two-step estimator, which will run substantially faster than continuous-updating. Set to FALSE to use the continuous-updating estimator described by Imai and Ratkovic (2014).

two.step Default is TRUE for a two-step estimator, which will run substantially faster than continuous-updating. Set to FALSE to use the continuous-updating estimator described by Imai and Ratkovic (2014).

sample.weights Survey sampling weights for the observations, if applicable. When left NULL, defaults to a sampling weight of 1 for each observation.

... Other parameters to be passed through to optim().

Value

CBPS.fit object

Description

hdCBPS high dimensional CBPS method to parses the formula object and passes the result to hdCBPS.fit, which calculates ATE using CBPS method in a high dimensional setting.

Usage

hdCBPS(
  formula,
  data,
  na.action,
  y,
  ATT = 0,
  iterations = 1000,
  method = "linear"
)

Arguments

formula An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted.

data An optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which CBPS is called.

na.action A function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset.
LaLonde

y
ATT
iterations
method

An outcome variable.
Option to calculate ATT
An optional parameter for the maximum number of iterations for the optimization. Default is 1000.
Choose among "linear", "binomial", and "poisson".

Value

ATT
ATE
s
fitted.values
coefficients1
coefficients0
model

Average treatment effect on the treated.
Average treatment effect.
Standard Error.
The fitted propensity score
Coefficients for the treated propensity score
Coefficients for the untreated propensity score
The model frame

Author(s)

Sida Peng

LaLonde Data for Covariate Balancing Propensity Score

Description

This data set gives the outcomes as well as treatment assignments and covariates for the econometric evaluation of training programs in LaLonde (1986).

Format

A data frame consisting of 12 columns (including a treatment assignment vector) and 3212 observations.

Source

Data from the National Supported Work Study. A benchmark matching dataset. Columns consist of an indicator for whether the observed unit was in the experimental subset; an indicator for whether the individual received the treatment; age in years; schooling in years; indicators for black and Hispanic; an indicator for marriage status, one of married; an indicator for no high school degree; reported earnings in 1974, 1975, and 1978; and whether the 1974 earnings variable is missing. Data not missing 1974 earnings are the Dehejia-Wahba subsample of the LaLonde data. Missing values for 1974 earnings set to zero. 1974 and 1975 earnings are pre-treatment. 1978 earnings is taken as the outcome variable.
References

npCBPS
Non-Parametric Covariate Balancing Propensity Score (npCBPS) Estimation

Description
npCBPS is a method to estimate weights interpretable as (stabilized) inverse generalized propensity score weights, \( w_i = f(T_i)/f(T_i|X) \), without actually estimating a model for the treatment to arrive at \( f(T|X) \) estimates. In brief, this works by maximizing the empirical likelihood of observing the values of treatment and covariates that were observed, while constraining the weights to be those that (a) ensure balance on the covariates, and (b) maintain the original means of the treatment and covariates.

In the continuous treatment context, this balance on covariates means zero correlation of each covariate with the treatment. In binary or categorical treatment contexts, balance on covariates implies equal means on the covariates for observations at each level of the treatment. When given a numeric treatment the software handles it continuously. To handle the treatment as binary or categorical it must be given as a factor.

Furthermore, we apply a Bayesian variant that allows the correlation of each covariate with the treatment to be slightly non-zero, as might be expected in a given finite sample.

Estimates non-parametric covariate balancing propensity score weights.

### @aliases npCBPS npCBPS.fit

Usage
npCBPS(formula, data, na.action, corprior = 0.01, print.level = 0, ...)

Arguments
- **formula**: An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted.
- **data**: An optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which CBPS is called.
- **na.action**: A function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset.
- **corprior**: Prior hyperparameter controlling the expected amount of correlation between each covariate and the treatment. Specifically, the amount of correlation between the k-dimensional covariates, X, and the treatment T after weighting is assumed to have prior distribution MVN(0,\( \sigma^2 I_k \)). We conceptualize \( \sigma^2 \) as
a tuning parameter to be used pragmatically. It’s default of 0.1 ensures that the balance constraints are not too harsh, and that a solution is likely to exist. Once the algorithm works at such a high value of sigma^2, the user may wish to attempt values closer to 0 to get finer balance.

print.level Controls verbosity of output to the screen while npCBPS runs. At the default of print.level=0, little output is produced. It print.level>0, it outputs diagnostics including the log posterior (log_post), the log empirical likelihood associated with the weights (log_el), and the log prior probability of the (weighted) correlation of treatment with the covariates.

... Other parameters to be passed.

Value

weights The optimal weights
y The treatment vector used
x The covariate matrix
model The model frame
call The matched call
formula The formula supplied
data The data argument
log.p.eta The log density for the (weighted) correlation of the covariates with the treatment, given the choice of prior (corprior)
log.el The log empirical likelihood of the observed data at the chosen set of IPW weights.
eta A vector describing the correlation between the treatment and each covariate on the weighted data at the solution.
sumw0 The sum of weights, provided as a check on convergence. This is always 1 when convergence occurs unproblematically. If it differs from 1 substantially, no solution perfectly satisfying the conditions was found, and the user may consider a larger value of corprior.

Author(s)

Christian Fong, Chad Hazlett, and Kosuke Imai

References

Examples

```r
##Generate data
data(LaLonde)

## Restricted two only two covariates so that it will run quickly.
## Performance will remain good if the full LaLonde specification is used
fit <- npCBPS(treat ~ age + educ, data = LaLonde, corprior=.1/nrow(LaLonde))
plot(fit)
```

Description

npCBPS.fit

Usage

```r
npCBPS.fit(treat, X, corprior, print.level, ...)
```

Arguments

treat A vector of treatment assignments. Binary or multi-valued treatments should be factors. Continuous treatments should be numeric.

X A covariate matrix.

corprior Prior hyperparameter controlling the expected amount of correlation between each covariate and the treatment. Specifically, the amount of correlation between the k-dimensional covariates, X, and the treatment T after weighting is assumed to have prior distribution MVN(0,\sigma^2 I_k). We conceptualize \sigma^2 as a tuning parameter to be used pragmatically. It's default of 0.1 ensures that the balance constraints are not too harsh, and that a solution is likely to exist. Once the algorithm works at such a high value of \sigma^2, the user may wish to attempt values closer to 0 to get finer balance.

print.level Controls verbosity of output to the screen while npCBPS runs. At the default of print.level=0, little output is produced. It print.level>0, it outputs diagnostics including the log posterior (log_post), the log empirical likelihood associated with the weights (log_el), and the log prior probability of the (weighted) correlation of treatment with the covariates.

... Other parameters to be passed.
plot.CBMSM

Plotting CBPS Estimation for Marginal Structural Models

Description

Plots the absolute difference in standardized means before and after weighting.

Usage

```r
## S3 method for class 'CBMSM'
plot(x, covars = NULL, silent = TRUE, boxplot = FALSE, ...)
```

Arguments

- `x`: an object of class “CBMSM”.
- `covars`: Indices of the covariates to be plotted (excluding the intercept). For example, if only the first two covariates from `balance` are desired, set `covars` to `1:2`. The default is `NULL`, which plots all covariates.
- `silent`: If set to `FALSE`, returns the absolute imbalance for each treatment history pair before and after weighting. This helps the user to create his or her own customized plot. Default is `TRUE`, which returns nothing.
- `boxplot`: If set to `TRUE`, returns a boxplot summarizing the imbalance on the covariates instead of a point for each covariate. Useful if there are many covariates.
- `...`: Additional arguments to be passed to `plot`.

Details

Covariate balance is improved if the plot’s points are below the plotted line of y=x.

Value

The x-axis gives the imbalance for each covariate-treatment history pair without any weighting, and the y-axis gives the imbalance for each covariate-treatment history pair after CBMSM weighting. Imbalance is measured as the absolute difference in standardized means for the two treatment histories. Means are standardized by the standard deviation of the covariate in the full sample.

Author(s)

Marc Ratkovic and Christian Fong

See Also

- `CBMSM`, `plot`
plot.CBPS  
Plotting Covariate Balancing Propensity Score Estimation

Description

This function plots the absolute difference in standardized means before and after weighting. To access more sophisticated graphics for assessing covariate balance, consider using Noah Greifer’s cobalt package.

Usage

```r
## S3 method for class 'CBPS'
plot(x, covars = NULL, silent = TRUE, boxplot = FALSE, ...)
```

Arguments

- `x`: an object of class “CBPS” or “npCBPS”, usually, a result of a call to CBPS or npCBPS.
- `covars`: Indices of the covariates to be plotted (excluding the intercept). For example, if only the first two covariates from `balance` are desired, set `covars` to 1:2. The default is `NULL`, which plots all covariates.
- `silent`: If set to `FALSE`, returns the imbalances used to construct the plot. Default is `TRUE`, which returns nothing.
- `boxplot`: If set to `TRUE`, returns a boxplot summarizing the imbalance on the covariates instead of a point for each covariate. Useful if there are many covariates.
- `...`: Additional arguments to be passed to `plot`.

Details

The "Before Weighting" plot gives the balance before weighting, and the "After Weighting" plot gives the balance after weighting.

```r
### @aliases plot.CBPS plot.npCBPS
```

Value

For binary and multi-valued treatments, plots the absolute difference in standardized means by contrast for all covariates before and after weighting. This quantity for a single covariate and a given pair of treatment conditions is given by

\[
\sum_{i=1}^{n} w_i \cdot (T_i = 1) \cdot X_i - \sum_{i=1}^{n} w_i \cdot (T_i = 0) \cdot X_i
\]

For continuous treatments, plots the weighted absolute Pearson correlation between the treatment and each covariate. See https://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient#Weighted_correlation_coefficient.

Author(s)

Christian Fong, Marc Ratkovic, and Kosuke Imai.
See Also

CBPS, plot

---

**plot.CBPSContinuous**  
*Plot the pre-and-post weighting correlations between X and T*

**Description**

Plot the pre-and-post weighting correlations between X and T

**Usage**

```r
## S3 method for class 'CBPSContinuous'
plot(x, covars = NULL, silent = TRUE, boxplot = FALSE, ...)
```

**Arguments**

- `x`: an object of class “CBPS” or “npCBPS”, usually, a result of a call to CBPS or npCBPS.
- `covars`: Indices of the covariates to be plotted (excluding the intercept). For example, if only the first two covariates from balance are desired, set `covars` to 1:2. The default is `NULL`, which plots all covariates.
- `silent`: If set to `FALSE`, returns the imbalances used to construct the plot. Default is `TRUE`, which returns nothing.
- `boxplot`: If set to `TRUE`, returns a boxplot summarizing the imbalance on the covariates instead of a point for each covariate. Useful if there are many covariates.
- `...`: Additional arguments to be passed to balance.

---

**plot.npCBPS**  
*Calls the appropriate plot function, based on the number of treatments*

**Description**

Calls the appropriate plot function, based on the number of treatments

**Usage**

```r
## S3 method for class 'npCBPS'
plot(x, covars = NULL, silent = TRUE, ...)
```
Arguments

x  an object of class “CBPS” or “npCBPS”, usually, a result of a call to CBPS or npCBPS.
covars  Indices of the covariates to be plotted (excluding the intercept). For example, if only the first two covariates from balance are desired, set covars to 1:2. The default is NULL, which plots all covariates.
silent  If set to FALSE, returns the imbalances used to construct the plot. Default is TRUE, which returns nothing.
...  Additional arguments to be passed to balance.

print.CBPS

Print coefficients and model fit statistics

Description

Print coefficients and model fit statistics

Usage

## S3 method for class 'CBPS'
print(x, digits = max(3, getOption("digits") - 3), ...)

Arguments

x  an object of class “CBPS” or “npCBPS”, usually, a result of a call to CBPS or npCBPS.
digits  the number of digits to keep for the numerical quantities.
...  Additional arguments to be passed to summary.

summary.CBPS

Summarizing Covariate Balancing Propensity Score Estimation

Description

Prints a summary of a fitted CBPS object.

Usage

## S3 method for class 'CBPS'
summary(object, ...)

Arguments

object  an object of class “CBPS”, usually, a result of a call to CBPS.
...  Additional arguments to be passed to summary.
vcov.CBPS

Details

Prints a summary of a CBPS object, in a format similar to glm. The variance matrix is calculated from the numerical Hessian at convergence of CBPS.

Value

call
The matched call.
deviance.residuals
The five number summary and the mean of the deviance residuals.
coefficients
A table including the estimate for the each coefficient and the standard error, z-value, and two-sided p-value for these estimates.
J
Hansen’s J-Statistic for the fitted model.
Log-Likelihood
The log-likelihood of the fitted model.

Author(s)

Christian Fong, Marc Ratkovic, and Kosuke Imai.

See Also

CBPS, summary

vcov.CBPS

Calculate Variance-Covariance Matrix for a Fitted CBPS Object

Description

vcov.CBPS Returns the variance-covariance matrix of the main parameters of a fitted CBPS object.

Usage

## S3 method for class 'CBPS'
vcov(object, ...)

Arguments

object
An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted.

... Additional arguments to be passed to vcov.CBPS

Details

This is the CBPS implementation of the generic function vcov().
vcov_outcome

Description

vcov_outcome Returns the variance-covariance matrix of the main parameters of a fitted CBPS object.

This adjusts the standard errors of the weighted regression of Y on Z for uncertainty in the weights.

Usage

vcov_outcome(object, Y, Z, delta, tol = 10^(-5), lambda = 0.01)
vcov_outcome

Arguments

- **object**: A fitted CBPS object.
- **Y**: The outcome.
- **Z**: The covariates (including the treatment and an intercept term) that predict the outcome.
- **delta**: The coefficients from regressing Y on Z, weighting by the cbpsfit$weights.
- **tol**: Tolerance for choosing whether to improve conditioning of the "M" matrix prior to conversion. Equal to 1/(condition number), i.e. the smallest eigenvalue divided by the largest.
- **lambda**: The amount to be added to the diagonal of M if the condition of the matrix is worse than tol.

Value

A matrix of the estimated covariances between the parameter estimates in the weighted outcome regression, adjusted for uncertainty in the weights.

Author(s)

Christian Fong, Chad Hazlett, and Kosuke Imai.

References

Lunceford and Davididian 2004.

Examples

```r
###
### Example: Variance-Covariance Matrix
###

#Load the LaLonde data
data(LaLonde)
# Estimate CBPS via logistic regression
fit <- CBPS(treat ~ age + educ + re75 + re74 + I(re75==0) + I(re74==0),
           data = LaLonde, ATT = TRUE)
# Get the variance-covariance matrix.
vcov(fit)
```
vcov_outcome.CBPSContinuous

vcov_outcome

Description

vcov_outcome

Usage

## S3 method for class 'CBPSContinuous'
vcov_outcome(object, Y, Z, delta, tol = 10^(-5), lambda = 0.01)

Arguments

object  A fitted CBPS object.
Y  The outcome.
Z  The covariates (including the treatment and an intercept term) that predict the outcome.
delta  The coefficients from regressing Y on Z, weighting by the cbpsfit$weights.
tol  Tolerance for choosing whether to improve conditioning of the "M" matrix prior to conversion. Equal to 1/(condition number), i.e. the smallest eigenvalue divided by the largest.
lambda  The amount to be added to the diagonal of M if the condition of the matrix is worse than tol.

Value

Variance-Covariance Matrix for Outcome Model
Index

* datasets
  - Blackwell, 7
  - LaLonde, 21
  - AsyVar, 2
  - balance, 5
  - balance.CBPS, 6
  - balance.CBPSContinuous, 6
  - balance.npCBPS, 7
  - Blackwell, 7
  - CBIV, 8
  - CBMSM, 10, 25
  - CBMSM.fit, 13
  - CBPS, 14, 27, 29
  - CBPS.fit, 19
  - hdCBPS, 20
  - LaLonde, 21
  - npCBPS, 22
  - npCBPS.fit, 24
  - plot, 25, 27
  - plot.CBMSM, 12, 25
  - plot.CBPS, 26
  - plot.CBPSContinuous, 27
  - plot.npCBPS, 27
  - print.CBPS, 28
  - summary, 29
  - summary.CBPS, 17, 28
  - vcov, 30
  - vcov.CBPS, 29
  - vcov.CBPSContinuous, 30
  - vcov_outcome, 30
  - vcov_outcome.CBPSContinuous, 32