Package 'nonrandom'

July 2, 2014

Type Package

Title Stratification and matching by the propensity score

Version 1.42

Date 2014-04-04

Author Susanne Stampf [cre, aut]

Maintainer Susanne Stampf <susanne.stampf@usb.ch>

Description This package offers a comprehensive data analysis if
stratification and matching by the propensity score is done.
Several functions are implemented, starting from the selection
of the propensity score model up to estimating propensity score
based treatment or exposure effects. All functions can be applied separately as well as combined.

Depends lme4

Suggests colorspace

License GPL

LazyLoad yes

LazyData yes

BuildVignettes yes

NeedsCompilation no

Repository CRAN

Date/Publication 2014-04-05 08:06:21

R topics documented:

`nonrandom-package` *A tool for a comprehensive data analysis from observational studies if stratification, matching and covariate adjustment by the propensity score is desired.*

Description

nonrandom offers a comprehensive data analysis if stratification, matching and covariate adjustment by the propensity score should be applied. Several functions are implemented, starting from the selection of the propensity score model up to estimating propensity score based treatment effects. All functions can be applied separately as well as combined.

Details

Package:	nonrandom
Type:	Package
Version:	1.3
Date:	2012-10-05
License:	What license is it under?

The estimation of the propensity score

The propensity score is the conditional probability of receiving a certain treatment given patient's covariates. It is generally unknown and has to be estimated, e.g. using logistic regression. The selection of an appropriate propensity score model is mostly difficult. A measure describing the

extent to which a covariate is confounding the treatment effect on outcome is implemented in `relative.effect()`. `pscore` estimates the propensity score.

Propensity score methods

Propensity score methods aims to balance covariate distributions between treatment groups and allow for estimating marginal effects. Stratification (`ps.makestrata()`) and matching (`ps.match`)) by the estimated propensity score are the most popular methods to eliminate imbalances in covariate distributions. `ps.estimate` also offers covariate adjustment by the propensity score.

Balance checks

An important, but often neglected issue in propensity score based data analyses is the check of covariate distributions between treatment groups after stratification and matching. Both graphical checks (`dist.plot` and `plot.stdf`) as well as classical statistical tests and standardized differences (`ps.balance`) can be used to examine covariate distributions.

Effect estimation

After stratification and matching by the propensity score, treatment effects are estimated in the stratified an matched data, respectively. `ps.estimate` estimates those effects depending on the data structure. It offers an additional adjustment for residual imbalances in the stratified or matched data as well as the estimation of effect using traditional regression models including covariate adjustment by the propensity score.

Author(s)

Susanne Stampf

Maintainer: Susanne Stampf <susanne.stampf@usb.ch>

Examples

```
## data on quality of life
data(stu1)

## estimate relative effects for covariates 'tgr' and 'age' regarding
## the effect of treatment 'therapie' on response 'pst'
rel.eff <- relative.effect(data    = stu1,
                           formula = pst~therapie+tgr+age)


## estimate the propensity score
ps <- pscore(data    = stu1,
             formula = therapie~tgr+age)


## stratify data
strata <- ps.makestrata(object = ps)


## match data in a ratio of 1:1 with a caliper size of 0.5
```

```
match <- ps.match(object  = ps,
                  ratio   = 1,
                  caliper = 0.5,
                  givenTmatchingC = FALSE)


## graphical check of distribution of both covariates between
## treatment groups in the matched data
bal.plot1 <- dist.plot(object = strata,
                       sel     = c("tmass"))

bal.plot2 <- dist.plot(object    = match,
                       sel       = c("alter"),
                       plot.type = 2,
                       compare   = TRUE)


## calculate standardized differences of both covariates
## in case of matched data
bal.table <- ps.balance(object  = match,
                        sel     = c("tgr","age"),
                        method  = "stand.diff",
                        alpha   =  20)


## estimate propensity score based effects and in comparion the
## regression based treatment effect on response
ps.est <- ps.estimate(object = strata,
                      resp   = "pst",
                      regr   = pst~therapie+tgr+age)
```

`dist.plot` *Graphical balance check for covariate distributions in treatment groups*

Description

Plot covariate disitribution in treatment groups

Usage

```
dist.plot(object, sel=NULL, treat=NULL, stratum.index=NULL,
match.index=NULL, plot.type=1, compare=FALSE, cat.levels=2,
plot.levels=5, label.match=NULL, label.stratum=c("Stratum","Original"),
with.legend=TRUE, legend.title=NULL, legend.cex=0.9, myoma=c(3,2,2,2),
mymar=c(5,4,1,2), width=0.5, xlim=NULL, ylim=NULL, col=NULL, las=1,
font.main=2, font=1, main=NULL, main.cex=1.2, sub.cex=0.9,
bar.cex=0.8, ...)
```

Arguments

object	an object of class 'pscore', 'stratified.pscore', 'stratified.data.frame', 'matched.pscore', 'matched.data.frame', 'matched.data.frames' or a data frame. If object class is 'pscore', arguments stratum.index or match.index are ignored.
sel	a data frame or a vector of integers or strings indicating covariates to be plotted. The default is 'NULL', i.e. the complete data set is selected.
treat	an integer or a string describing the treatment indicator in 'data' and 'data.matched', respectively, if ps.match() is previously used. If the class of the input object is 'stratified.pscore' or 'matched.pscore', no specification is needed.
stratum.index	an integer or a string indicating the vector containing the stratum indices in stratified data. No specification is needed if ps.makestrata() is previously used.
match.index	an integer or a string indicating the vector containing the matching indices in data and in the matched data. No specification is needed if ps.match() is previously used.
plot.type	an integer specifying the plot type. The default is '1', i.e. means for continuous and frequencies for categorical covariates are plotted as barplots separated by treatment. If plot.type='2', histograms are shown.
compare	a logical value indicating whether the covariate distribution in the original data are plotted.
cat.levels	an integer. The default is '2', i.e. covariates with more than two different values are considered as continuous.
plot.levels	an integer. The default is '5', i.e. five cutpoints are used to define histogram classes for continuous covariates. Caution: The classification depends on the data structure such that the class number used in the histogram may differ from the statement in plot.levels.
label.match	a vector of two strings describing the labels for the original and the matched data. The default is 'NULL', i.e. c('Original', 'Matched') is used.
label.stratum	a string describing the labels for the stratum-specific data.
with.legend	a logical value indicating whether a legend is shown.
legend.title	a string indicating the legend title. The default is 'NULL', i.e either covariate categories or treatment labels in case of continuous covariates are given if plot.type='1'. For plot.type='2', treatment labels are shown.
legend.cex	a numeric indicating the font size in the legend.
myoma	the size of outer margins, see par.
mymar	margins to be specified on the four sides of the plot, see par.
width	an integer indicating bar widths.
xlim	a vector of integers of length two indicating limits for the x axis.
ylim	a vector of integers of length two indicating limits for the y axis. It is only meaningful if plot.type='2'.
col	a vector of colors for bars or bar components. The vector length should depend on the plot.type and the category levels of the plotted covariates.

las	a integer indicating the style of axis labels, see par.
font.main	an integer indicating the font to be used for plot main titles, see par.
font	an integer specifying the font to use for text, see par.
main	a string indicating the main title for graphics.
main.cex	a numeric indicating the font size of main title.
sub.cex	a numeric indicating the font size of sub titles.
bar.cex	a numeric indicating the font size of bar titles.
...	further arguments for graphics.

Details

Propensity score methods aims to eliminate imbalances in covariate distributions between treatment groups. An important issue is to check those after stratification or matching.

The usage of dist.plot() depends on the class of the input object. If either ps.makestrata() or ps.match() are previously used, treat, match.index and stratum.index are not needed, contrary to the case where the input object is a data frame.

Value

dist.plot() returns a list containing information for graphics. The number and the manner of the list entries depends on plot.type and on the type of covariates to be plotted:

name.sel	a string containing names of the selected covariates.
sel	a data frame containing the selected covariates labeled by 'name.sel'.
name.treat	a string indicating the name of the selected treatment variable.
treat	a vector containing the treatment variable labeled by 'name.treat'.
name.stratum.index	a string indicating the name of the selected stratum indices.
stratum.index	a vector containing the stratum variable labeled by 'name.stratum.index'.
name.match.index	a string indicating the name of the selected matching indices.
match.index	a vector containing the matching variable labeled by 'name.match.index'.
var.cat	a string indicating the names of categorical variables.
var.noncat	a string indicating the names of continuous variables.
mean	a list of length two including means of continuous covariates separated by treatment. If compare='FALSE', list elements are matrices with means separated by treatment (rows) and strata or matched/unmatched data (columns), respectively. If compare='TRUE', the list elements are lists including means before (first list element) and after (second list element) stratification or matching. The order of list elements is in accordance to the order of 'var.noncat'. It is only available if plot.type=1.

`frequency` a list with length according to the number of categorical covariates whereas the list elements depend on `compare`. If `compare='FALSE'`, list elements contain standardized frequency tables separated by treatment (rows) and by strata or by matched/unmatched data (columns). The order and the number of list elements is w.r.t. 'var.cat'. If `compare='TRUE'`, there are two list elements which are lists of frequency tables. The first list element contains lists of frequency tables separated by treatment before stratification or matching and the second list element includes lists of frequency tables separated by treatment and strata or matched/unmatched data. The order of list elements is in accordance to the order of 'var.cat'. It is only available if `plot.type=1`.

`breaks.noncat` a list with length according to the number of continuous covariates. The list entries are numerics indicating the cutpoints of histogram classes. It is only available if `plot.type=2`.

`x.cat` a list with length according to the number of categorical covariates. It contains frequencies w.r.t. treatment with the lower value, e.g., '0' or 'No', before stratification or matching. It is only available if `plot.type=2` and `compare='TRUE'`.

`y.cat` a list with length according to the number of categorical covariates. It contains frequencies w.r.t. treatment with the upper value, e.g., '1' or 'Yes', before stratification or matching. It is only available if `plot.type=2` and `compare='TRUE'`.

`x.s.cat` a list with length according to the number of categorical covariates. It contains frequencies w.r.t. strata (columns) and treatment with the lower value, e.g., '0' or 'No' after stratification. If `match.index` is used, frequencies are given w.r.t. treatment with the lower value and w.r.t. the original data (first column) and the matched data (second column). It is only available if `plot.type=2`.

`y.s.cat` a list with length according to the number of categorical covariates. It contains frequencies w.r.t. strata (columns) and treatment with the upper value, e.g., '1' or 'Yes' after stratification. If `match.index` is used, frequencies are given w.r.t. treatment with the upper value and w.r.t. the original data (first column) and the matched data (second column). It is only available if `plot.type=2`.

`x.noncat` a list with length according to the number of continuous covariates. It contains frequencies in histogram classes w.r.t. treatment with the lower value, e.g., '0' or 'No', before stratification or matching. It is only available if `plot.type=2` and `compare='TRUE'`.

`y.noncat` a list with length according to the number of continuous covariates. It contains frequencies in histogram classes w.r.t. treatment with the upper value, e.g., '1' or 'Yes', before stratification or matching. It is only available if `plot.type=2` and `compare='TRUE'`.

`x.s.noncat` a list with length according to number of continuous covariates. It contains lists with frequencies in histogram classes w.r.t. strata and treatment with the lower value, e.g., '0' or 'No', after stratification. If `match.index` is used, frequencies in histogram classes are given w.r.t. treatment with the lower value and w.r.t. the original data and the matched dats. It is only available if `plot.type=2`.

`y.s.noncat` a list with length according to number of continuous covariates. It contains lists with frequencies in histogram classes w.r.t. strata and treatment with the upper value, e.g., '1' or 'Yes', after stratification. If `match.index` is used, frequencies in histogram classes are given w.r.t. treatment with the upper value and w.r.t. the original data and the matched dats. It is only available if `plot.type=2`.

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

See Also

barplot

Examples

```
## STU1
data(stu1)
stu1.ps <- pscore(data    = stu1,
                  formula = therapie~tgr+age)
stu1.match <- ps.match(object          = stu1.ps,
                       ratio           = 2,
                       caliper         = 0.5,
                       givenTmatchingC = FALSE,
                       matched.by      = "pscore",
                       setseed         = 38902)
stu1.plot <-
    dist.plot(object      = stu1.match,
              sel         = c("age"),
              compare     = TRUE,
              plot.type   = 2,
              with.legend = FALSE)

## PRIDE
data(pride)
pride.ps <- pscore(data        = pride,
                   formula     = PCR_RSV~SEX+RSVINF+REGION+
                                 AGE+ELTATOP+EINZ+EXT,
                   name.pscore = "ps")
pride.strata <- ps.makestrata(object = pride.ps,
                              breaks = quantile(pride.ps$pscore,
                                                seq(0,1,0.2)),
                              stratified.ps = "ps")
pride.plot <-
   dist.plot(object    = pride.strata,
             sel       = c("REGION", "AGE"),
             plot.type = 1)                 ## default
```

`plot.pscore` *Graphical check for propensity score distributions in treatment groups*

Description

Plot propensity score density in treatment groups

Usage

```
## S3 method for class 'pscore'
plot(x, par.dens=NULL, par.1=NULL, par.0=NULL,
  with.legend=FALSE, legend.cex=0.9, legend.label=NULL,
  main=NULL, ylim=NULL, xlim=NULL, ...)
```

Arguments

x	an object of class 'pscore'.
par.dens	a list of parameters needed for function density() internally used. The default is NULL, i.e. default parameters in density() are used.
par.1	a list of parameters needed for function lines() internally used for the presentation of the propensity score density for individuals labeled by treat='1'. The default is NULL, i.e. default parameter in lines() are used.
par.0	a list of parameters needed for function lines() internally used for the presentation of the propensity score density for individuals labeled by treat='0'. The default is NULL, i.e. 'lty=2' and remaining default parameter in lines() are used.
with.legend	a logical value for showing a legend.
legend.cex	a numeric value indicating the cex of the legend font.
legend.label	a vector of two strings labeling treated and untreated individuals. The default is NULL, i.e. c('treated', 'untreated') is used.
main	a string indicating the title of the plot.
ylim	a vector of two numerics indicating the limits of the y-axis.
xlim	a vector of two numerics indicating the limits of the x-axis.
...	further arguments for graphics.

Details

Propensity score methods aim to eliminate imbalances in covariate distributions between treatment groups. Therefore, individuals from both treatments are matched together or individuals are stratified based on their propensity score. To do so, the graphical check of propensity score distributions for treated and untreated individuals is useful.

The use of plot.pscore() requires the use of (pscore).

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

See Also

plot

Examples

```
## STU1
data(stu1)
stu1.ps <- pscore(data    = stu1,
                  formula = therapie~tgr+age)

plot.pscore(x    = stu1.ps,
            main = "PS distribution",
            xlab = "",
            par.1=list(col="red"),
            par.0=list(lwd=2),
            par.dens=list(kernel="gaussian"))
```

`plot.stdf` *Graphical check for standardized differences of covariates*

Description

Plot standardized differences of covariates

Usage

```
 ## S3 method for class 'stdf'
plot(x, sel = NULL, plot.alpha = TRUE,
  mymar = c(5,8,4,2), pch.p = c(1,5), col.p = c("black", "red"),
  colorspace = NULL, cex.p = 1.25, line.stdf = 1, line.alpha = 4,
  with.legend = TRUE, legend.label = c("before", "after"),
  legend.cex = 1, legend.xy = NULL, ...)
```

Arguments

`x`	an object of class 'bal.matched.data.frame', 'bal.matched.data.frames' or 'bal.matched.pscore'. The previous use of `ps.balance(..., method='stand.diff', ...)` is needed.
`sel`	a vector of strings indicating covariates to be checked. The default is NULL, i.e. all variables selected previously in `ps.balance()` are plotted.
`plot.alpha`	a logical value indicating whether a vertical line for the significance level chosen in `ps.balance()` should be plotted.
`mymar`	a graphical parameter, see `par()`.
`pch.p`	a vector of two integers indicating the symbols.
`col.p`	a vector of two strings indicating symbol colors.
`colorspace`	a logical value indicating whether `rainbow_hcl(20)` (if TRUE) or `grey.colors(20)` (if FALSE) should be used (argument '20' indicate that 20 colors are pre-specified and can be selected using `col.p`). The default is NULL, i.e. argument `col.p` is used. The colors are randomly chosen when argument `col.p` contains strings and `colorspace` is set to TRUE.

cex.p	a numeric indicating the size of symbols.
line.stdf	an integer indicating the line type connecting the sympols.
line.alpha	an integer indicating the line type presenting the significance level.
with.legend	a logical value indicating whether the legend is given.
legend.label	a vector of two strings indicating the labels.
legend.cex	a numeric indicating the size of the legend font.
legend.xy	a vector of two integers indicating x- and y-coordinates for legend position.
...	further arguments for graphics.

Details

Standardized difference are proposed to check the balance of covariates after matching. The function plot.stdf() offers a graphical check of it presenting standardized differences of covariates between treatment groups before and after matching.

The usage of plot.stdf() requires the previous use of ps.balance(..., method='stand.diff', ...) and is only available when matching is done before via ps.match().

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

See Also

plot

Examples

```
## STU1
data(stu1)
stu1.ps <- pscore(data    = stu1,
                  formula = therapie~tgr+age)

stu1.match <- ps.match(object          = stu1.ps,
                       ratio           = 2,
                       caliper         = 0.5,
                       givenTmatchingC = FALSE,
                       matched.by      = "pscore",
                       setseed         = 38902)

stu1.bal.match <- ps.balance(object = stu1.match,
                             sel    = c("tgr","age"),
                             method = "stand.diff",
                             alpha  = 20)

plot.stdf(x          = stu1.bal.match,
          main       = "Stu1 study: Standardized differences",
          cex.axis   = 1.2,
          legend.cex = 1.3,
          cex.main   = 1.5,
```

```
las       = 1,
col.p     = c("black", "gray"))
```

`pride` *Data example pride*

Description

Data dealing with lower respiratory tract infections (LRTI) in n=3.078 infants and children aged less than three years in the observational study Pri.DE (Pediatric Respiratory Infection, Deutschland) in Germany.

Usage

```
data(pride)
```

Format

A data frame with n=3078 observations on the following 15 variables.

PCR_RSV current respiratory syncytial virus (RSV) infection (1) or not (0)

SEX gender: male (1) and female (2)

ETHNO ethnic group: German (1), European Union (2) and others (3)

FRUEHG preterm delivery (1) or not (0)

RSVINF former RSV infection (1) or not (0)

HERZ congenital heart defect (1) or not (0)

REGION German region: South (1), East (2), West (3) and North (4)

AGE age in years

VOLLSTIL current breast feeding or longer than two months (1) or not (0)

EINZ siblings (1) or not (0)

TOBACCO passive tobacco smoke exposure at home (1) or not (0)

EXT external care (1) or not (0)

ELTATOP parental atopy (1) or not (0)

SEVERE severe LRTI (1) or not (0)

KRANKSUM number of diagnosed LRTI

Examples

```
data(pride)
```

print.relative.effect *Print function*

Description

to print results.

Usage

```
 ## S3 method for class 'relative.effect'
print(x, ...)
```

Arguments

x	an object to be printed.
...	further arguments passed to.

Details

Print function

ps.balance *Statistical tests and standardized differences for balance checks*

Description

Apply statistical tests or calculate standardized differences for balance checks of covariate distributions between treatment groups

Usage

```
ps.balance(object, sel=NULL, treat=NULL, stratum.index=NULL,
match.index=NULL,method="classical", cat.levels=2, alpha=5,
equal=TRUE)
```

Arguments

object	an object of class 'stratified.pscore', 'stratified.data.frame', 'matched.pscore', 'matched.data.frame', 'matched.data.frames' or a data frame.
sel	a data frame or a vector of integers or strings indicating covariates to be checked. The default is 'NULL', i.e. all variables in the data are selected.

`treat` an integer or a string indicating the treatment variable in the data (and in the matched data if `ps.match()` is previously used). If the class of the input object is 'stratified.pscore' or 'matched.pscore', no specification is needed.

`stratum.index` an integer or a string indicating the vector containing the stratum indices in the stratified data. No specification is needed if `ps.makestrata()` is previously used.

`match.index` an integer or a string indicating the vector containing matching indices in data and in matched data. No specification is needed if `ps.match()` is previously used.

`method` a string indicating the method used to decide about covariate balance between treatment groups. The default is 'classical', i.e., `ttest()` for continuous and `chisq.test()` for categorical covariates are applied, both in original data and in stratified or matched data. If 'stand.diff', standardized differences are calculated for covariates before and after stratification or matching (formulas are nicely explaind in: D. Yang and J.E. Dalton: 'A unified approach to measuring the effect size between two groups using SAS', Paper 335-2012, SAS Global Forum 2012: Statistics and Data Analysis).

`cat.levels` an integer indicating the maximal number of levels of selected categorical covariates to consider them as categorical. The default is '2', i.e., covariates with more than two different values are considered as continuous. For example, cov1 and cov2 has three and four levels, respectively. `cat.levels` should be set to 4 to consider both as categorical. Caution: If covariates are factors and `cat.levels` is not appropriately chosen, errors can occur since `t.test` can not be performed!

`alpha` an integer indicating the significance level (per cent) or the cutpoint at which the decision about balance or imbalance is made in case of standardized differences.

`equal` a logical value. The default is 'TRUE', i.e. equally-sized weights are used to combine standard deviations of covariates in treatment groups for calculating standardized differences. If 'FALSE', weights are proportions of observations in treatment groups within data, matched data and strata.

Details

Propensity score methods aims to eliminate imbalances in covariate distributions between treatment groups. An important issue is to check those after stratification or matching. Statistical tests or standardized differences can be used for those balance checks.

The usage of `ps.balance()` depends on the class of the input object. If either `ps.makestrata()` or `ps.match()` are previously used, `treat`, `match.index` and `stratum.index` are not needed, contrary to the case where the input object is a data frame.

Value

`ps.balance()` returns an object of the same class as the input object. The number and the manner of the values depends on the used `method`:

`data` a data frame containing the input data.

`data.matched` a data frame limiting 'data' only to matched observations. It is only available if `ps.match()` is previously used.

`name.stratum.index` a string indicating the name of the selected stratum variable. It is only available if `ps.makestrata()` is previously used.

`stratum.index` a numeric vector containing stratum indices labeled by 'name.stratum.index'. It is only available if `ps.makestrata()` is previously used.

`intervals` a vector of characters indicating intervals. It is only available if `ps.makestrata()` is previously used.

`stratified.by` a string indicating the name of stratification variable. It is only available if `ps.makestrata()` is previously used.

`formula.pscore` a formula describing formally the propensity score model fitted at last in `pscore()` .

`model.pscore` an object of class `glm` containing information about the propensity score model fitted at last in `pscore()`.

`name.pscore` a string indicating the name of propensity score at last estimated via `pscore()` and saved in the data.

`pscore` a numeric vector containing the estimated propensity score labeled by 'name.pscore'.

`name.treat` a string indicating the name of the treatment variable.

`treat` a numeric vector containing the treatment variable labeled by 'name.treat'.

`matched.by` a string indicating the name of the matching variable. It is only available if `ps.match()` is previously used.

`name.match.index` a string indicating the name of the selected matching variable. It is only available if `ps.match()` is previously used.

`match.index` a numeric vector containing the matching indices labeled by 'name.match.index' whereas '0' indicates 'no matching partner found'. It is only available if `ps.match()` is previously used.

`match.parameters` a list of matching parameters including `caliper`, `ratio`, `who.treated`, `givenTmatchingC` and `bestmatch.first`. It is only available if `ps.match()` is previously used.

`bal.test` a list of elements describing the results for the performed balance checks.

- **balance.table** a 2xK table describing the balance of K covariate distributions between treatment groups. The first/second row presents results from balance checks before/after stratification or matching. '0'/'1' indicates significant/non-significant differences between treatment groups.
- **balance.table.summary** a 2x2 table summarizing balance information from 'balance.table' across all K covariates. Only variables with balance tests done correctly are accounted for this table.
- **covariates.NA** a vector of strings indicating the names of covariates for which balance checks could not be done correctly.
- **covariates.bal.before** a vector of strings indicating the names of covariates balanced before stratification or matching.

covariates.bal.after a vector of strings indicating the names of covariates balanced after stratification or matching.

p.value a (s+1)xK table containing p-values from statistical tests for K covariates before (first row) and after (2nd, ..., (s+1)st row) stratification or matching (s is the number of defined strata; if matching, s=1). It is only available if `method='classical'`.

Stand.diff a 2xK or (s+1)xK table containing standardized differences for K covariates before (first row) and after matching (2nd row) or stratification (2nd, ..., (s+1)st row). It is only available if `method='stand.diff'`.

method a vector of strings indicating the scale of covariates assumed for balance checks ('none', 'bin', 'cat' or 'num'). The value 'none' means that no balance check was performed and 'bin', 'cat' and 'num' indicate binary, categorical and continuous.

alpha a numeric value defining the significance level or the cut point at which the decision about balance or imbalance is made.

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

Examples

```
## STU1
data(stu1)
stu1.ps <- pscore(data    = stu1,
                  formula = therapie~tgr+age)
stu1.match <- ps.match(object          = stu1.ps,
                       ratio           = 2,
                       caliper         = 0.5,
                       givenTmatchingC = FALSE,
                       matched.by      = "pscore",
                       setseed         = 38902)
stu1.balance <- ps.balance(object  = stu1.match,
                           sel     = c("tgr","age"),
                           method  = "stand.diff",
                           alpha   =  20)

## PRIDE
data(pride)
pride.ps <- pscore(data        = pride,
                   formula     = PCR_RSV~SEX+RSVINF+REGION+
                                 AGE+ELTATOP+EINZ+EXT,
                   name.pscore = "ps")
pride.strata <- ps.makestrata(object = pride.ps,
                              breaks = quantile(pride.ps$pscore,
                                                seq(0,1,0.2)),
                              stratified.ps = "ps")
pride.balance <- ps.balance(object     = pride.strata,
                            sel        = c(2:6),
                            method     = "classical",
                            cat.levels = 4,
```

```
                    alpha      = 5)
```

---

ps.estimate *Estimation of propensity score based treatment effects*

Description

Estimation of propensity score based treatment effects

Usage

```
ps.estimate(object,resp,treat = NULL, stratum.index = NULL,
match.index = NULL, adj = NULL, weights = "rr", family = "gaussian",
regr = NULL, ...)
```

Arguments

object	an object of class 'stratified.pscore', 'stratified.data.frame', 'matched.pscore', 'matched.data.frame', 'matched.data.frames' or a data frame.
resp	an integer or a string indicating the outcome variable in the data and in the matched data if ps.match() is previously used.
treat	an integer or a string indicating the treatment variable in the data and in the matched data if ps.match() is previously used. If the class of the input object is 'stratified.pscore' or 'matched.pscore', no specification is needed.
stratum.index	an integer or a string indicating the vector containing the stratum indices. No specification is needed if ps.makestrata() is previously used.
match.index	an integer or a string indicating the vector containing the matching indices. No specification is needed if ps.match() is previously used.
adj	a formula or a vector of integers or strings indicating covariates. The default is NULL, i.e. no additional adjustment for covariates is done in stratified or matched data. If adj is a formula, it must be formulated as 'outcome~treatment+covariates' according to resp and treat. If adj is a vector, it contains names or integers indicating covariates for which the treatment effect has to be adjusted for.
weights	a string indicating how to weight the stratum-specific estimates to obtain an overall estimate in case of stratified data. The default is 'rr' indicating stratum-specific weights proportional to the stratum-specific number of observations. In case of linear outcome, weights = 'opt' can be used, i.e. weights are optimal in sence of variance minimizing.
family	a description of the error distribution and link function to be used in the model (see glm).

regr	a formula or a vector of integers or strings indicating covariates in data. The default is NULL, i.e. no regression model is fitted. If regr is a formula, it must be formulated as 'outcome~treatment+covariates' according to resp and treat. If regr is a vector, it contains names or integers indicating covariates for which the treatment effect on outcome estimated by regression has to be adjusted.
...	further arguments passed to or from other methods.

Details

Propensity score methods are used to estimate treatment effects in observational data. The treatment effects are estimated without adjustment and can to be interpreted as marginal effects. However, it is additionally possible to adjust for residual imbalances in strata or in the matched data (using adj) and also to apply traditional regression (using regr).

The usage of ps.estimate() depends slightly on the class of the input object. If either ps.makestrata() or ps.match() are previously used, treat, match.index and stratum.index are not needed, contrary to the case where the input object is a data frame. If both match.index and stratum.index are specified, stratum.index will be ignored. If one specifies adj and regr as formulas, they must be identical in form of 'outcome~treatment+covariates' and 'outcome' and 'treatment' must agree with resp and treat.

Value

ps.estimate returns an object of the same class as the input object. The number and the manner of values depends on the scale of resp:

data	a data frame containing the input data.
data.matched	a data frame limiting 'data' only to matched observations. It is only available if ps.match() is previously used.
name.resp	a string indicating the name of the outcome variable.
resp	a numeric vector indicating the outcome variable labeled by 'name.resp'.
name.stratum.index	a string indicating the name of the selected stratum indices. It is only available if ps.makestrata() is previously used.
stratum.index	a numeric vector containing the selected stratum indices labeled by 'name.stratum.index'. It is only available if ps.makestrata() is previously used.
intervals	a vector of characters indicating the interval used for stratification. It is only available if ps.makestrata() is previously used.
stratified.by	a string indicating the name of the selected stratification variable. It is only available if ps.makestrata() is previously used.
formula.pscore	a formula describing formally the propensity score model fitted at last.
model.pscore	an object of class glm containing information about the propensity score model fitted at last.
name.pscore	a string indicating the name of propensity score estimated at last.
pscore	a numeric vector containing the estimated propensity score labeled by 'name.pscore.
name.treat	a string indicating the name of the treatment variable.

`treat` a numeric vector containing the treatment index labeled by 'name.treat'.

`matched.by` a string indicating the name of the selected matching variable. It is only available if `ps.match()` is previously used.

`name.match.index` a string indicating the name of the selected matching indices. It is only available if `ps.match()` is previously used.

`match.index` a numeric vector containing the selected matching indices labeled by 'name.match.index' whereas '0' indicates 'no matching partner found'. It is only available if `ps.match()` is previously used.

`match.parameters` a list of matching parameters including `caliper`, `ratio`, `who.treated`, `givenTmatchingC` and `bestmatch.first`. It is only available if `ps.match()` is previously used.

`lr.estimation` a list containing information about the regression model based treatment effect estimates. It correspond to the argument `regr`.

- **effect** the estimated conditional treatment effect based on regression. It is an odds ratio for binary outcome.
- **effect.marg** the estimated marginal treatment effect based on regression. It is an odds ratio for binary outcome. Conditional and marginal treatment effects are identical if the outcome is assumed to be normally distributed.
- **se** the estimated standard error for `effect`. For binary outcome, it is given on the log scale.
- **se.marg** the estimated standard error for `effect.marg`. For binary outcome, it is given on the log scale.
- **regr.formula** a formula describing formally the fitted regression outcome model correspond to `regr`.
- **regr.model** a `glm` object containing information of the fitted regression outcome model correspond to `regr`.

`ps.estimation` a list containing information about the estimated propensity score based treatment effects. The list entries depend on the scale of outcome.

- **crude** a list containing information about the crude treatment effect estimated by 'outcome~treatment':
  - **effect** the estimated crude treatment effect. For binary outcome, it is an odds ratio.
  - **se** the estimated standard error of 'effect'. For binary outcome, it is given on log scale.
- **adj** a list containing information about the estimated adjusted propensity score based treatment effect corresponding to `adj`:
  - **model** outcome model applied in each stratum or in the matched data to adjusted for covariates.
  - **effect.str** the estimated adjusted stratum-specific treatment effects. For binary outcome, it is an odds ratio. It is only availabe for stratified data.
  - **effect** the estimated adjusted overall treatment effect. For binary outcome, it is an odds ratio.
  - **se** the estimated standard error of `effect`. For binary outcome, it is given on log scale.

unadj a list containing information about the estimated unadjusted propensity score based treatment effect. The list entries depend on the scale of response:

- **effect** the estimated marginal treatment effect. For binary outcome, it is an odds ratio based on outcome rates.
- **se** the estimated standard error of `effect`. For binary outcome, it is given on log scale.
- **p1** a numeric indicating the estimated marginal outcome probability for treatment labeled by the upper value. It is only available for binary outcome and stratified data.
- **p0** a numeric indicating the estimated marginal outcome probability for treatment labeled by the lower value. It is only available for binary outcome and stratified data.
- **p1.str** a numeric indicating the estimated stratum-specific outcome probabilities for treatment labeled by the upper value. It is only available for binary outcome and stratified data.
- **p0.str** a numeric indicating the estimated stratum-specific outcome probabilities for treatment labeled by the lower value. It is only available for binary outcome and stratified data.
- **effect.mh** the stratified Mantel-Haenszel estimate for the treatment effect. It is only available for binary outcome and stratified data.
- **se.mh** the estimated standard error for `effect.mh`. It is only available for binary outcome and stratified data.
- **odds.str** the estimated stratum-specific odds ratio used for the stratified Mantel-Haenzsel estimator. It is only available for binary outcome and stratified data.

weights a string indicating the selected weights scheme.

weights.str a numeric vector containing the stratum-specific weights. It is only available for stratified data.

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

See Also

`glm`, formula, `lmer`

Examples

```
## STU1
data(stu1)
stu1.ps <- pscore(data    = stu1,
                  formula = therapie~tgr+age)
stu1.match <- ps.match(object          = stu1.ps,
                       ratio           = 2,
                       caliper         = 0.5,
                       givenTmatchingC = FALSE,
```

```
                       matched.by      = "pscore",
                       setseed         = 38902)
stu1.est <-
  ps.estimate(object  = stu1.match,
              resp    = "pst",
              adj     = "tmass",
              regr    = pst~therapie+tgr+age)

## PRIDE
data(pride)
pride.ps <- pscore(data        = pride,
                   formula     = PCR_RSV~SEX+RSVINF+REGION+
                                 AGE+ELTATOP+EINZ+EXT,
                   name.pscore = "ps")
pride.strata <- ps.makestrata(object = pride.ps,
                              breaks = quantile(pride.ps$pscore,
                                                seq(0,1,0.2)),
                              stratified.ps = "ps")
pride.est <-
  ps.estimate(object  = pride.strata,
              resp    = "SEVERE",
              family  = "binomial",
              adj     = "AGE",
              regr    = SEVERE ~PCR_RSV+SEX+ETHNO+FRUEHG+HERZ+
                        ELTATOP+REGION+AGE+TOBACCO+VOLLSTIL+EXT+
                        EINZ+KRANKSUM,
              weights = "rr")
```

`ps.makestrata` *Propensity score stratification*

Description

Stratification based on the estimated propensity score

Usage

```
ps.makestrata(object, breaks=NULL, name.stratum.index="stratum.index",
stratified.by=NULL, ...)
```

Arguments

`object`	an object of class 'pscore' or a data frame.
`breaks`	an integer, a numeric vector or suitable R function, e.g `quantile()`. The default is 'NULL', i.e. stratum bounds are automatically determined (see `cut()`).
`name.stratum.index`	a string indicating the variable name containing the stratum indices.

stratified.by	an integer or a string indicating the stratification variable in data. The default is NULL, i.e. if the class of the input object is 'pscore', object value 'pscore' is automatically used.
...	further arguments passed to or from other methods.

Details

Stratification by the estimated propensity score groups observations with identical or similar estimated propensity score.

If function pscore() is previously used with default settings, stratified.by has not to be specified. It is needed if the stratification variable is not labeled by 'pscore'.

Several options for the argument breaks are available. The default is 'NULL', i.e. values of the stratification variable are factorized and each factor correspond to a stratum. Using an integer, the number of strata is specified. If a numeric vector is given, its values indicate the stratum bounds.

Value

ps.makestrata() returns an object of class 'stratified.pscore' or 'stratified.data.frame' depending on the class of the input object. If the class of the input object is 'pscore', the output object inherits all components from the input object. The following components are available:

data	a data frame containing the input data, extended by column(s) including stratum indices labeled by name.stratum.index.
name.stratum.index	a string indicating the name of the variable containing the stratum indices, generated at last.
stratum.index	a numeric vector containing the stratum indices labeled by 'name.stratum.index'.
intervals	a vector of characters indicating intervals corresponding to stratum.index.
stratified.by	a string indicating the name of the stratification variable.
formula.pscore	a formula describing formally the propensity score model fitted at last in pscore().
model.pscore	an object of class glm containing information about the propensity score model fitted at last in pscore().
name.pscore	a string indicating the name of the propensity score estimated at last in pscore().
pscore	a numeric vector containing the estimated propensity score labeled by 'name.pscore'.
name.treat	a string indicating the name of treatment used.
treat	a numeric vector containing treatment labeled by 'name.treat'.

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

See Also

cut, quantile

Examples

```
## STU1
data(stu1)
stu1.ps <- pscore(data    = stu1,
                  formula = therapie~tgr+age)
stu1.strata <- ps.makestrata(object = stu1.ps)


## PRIDE
data(pride)
pride.ps <- pscore(data        = pride,
                   formula     = PCR_RSV~SEX+RSVINF+REGION+
                                 AGE+ELTATOP+EINZ+EXT,
                   name.pscore = "ps")
pride.strata <- ps.makestrata(object = pride.ps,
                              breaks = quantile(pride.ps$pscore,
                                                seq(0,1,0.2)),
                              stratified.ps = "ps")
```

`ps.match` *Propensity score matching*

Description

Matching based on the estimated propensity score

Usage

```
ps.match(object, object.control=NULL, matched.by=NULL,
control.matched.by=matched.by, who.treated=1, treat=NULL,
name.match.index="match.index", ratio=1, caliper="logit", x=0.2,
givenTmatchingC=TRUE, bestmatch.first=TRUE, setseed=FALSE,
combine.output=TRUE)
```

Arguments

`object`	an object of class 'pscore' or a data frame.
`object.control`	a data frame. It is needed if `object` is a data frame including only treated or untreated observations, respectively.
`matched.by`	an integer or a string indicating the matching variable. The default is NULL, i.e. if the class of the input object is 'pscore', object value 'pscore' is automatically used.
`control.matched.by`	an integer or a string indicating the matching variable in `object.control`. The default is 'matched.by'.

`who.treated` an integer or a string indicating which value of `treat` labels the 'treated' observations.

`treat` an integer or a string indicating the treatment variable in data both of `object` and `object.control`. If the class of the input object is 'pscore', no specification is needed.

`name.match.index` a string indicating the name of the variable containing the matching indices.

`ratio` an integer k indicating the matching ratio 1:k.

`caliper` an integer or a string indicating the maximum width of the caliper for which matching should be done. The default is 'logit', i.e. the maximum width of the caliper is `x` of the standard deviation of the logit of the matching variable.

`x` a numeric value indicating the scale parameter for the calculation of the caliper if `caliper='logit'`.

`givenTmatchingC` a logical value indicating who is matched to whom. The default is 'TRUE', i.e. untreated observations are matched to treated observations.

`bestmatch.first` a logical value indicating how potential matching partners are matched. The default is 'TRUE', i.e. observations are matched with the best accordance regarding the matching variable. Otherwise, matching partners are randomly assigned from the pool of potential matching candidates.

`setseed` an integer setting a random number for the matching process.

`combine.output` a logical value. The default is 'TRUE', i.e. if `object` and `object.control` are given, 'data', 'data.matched' and 'match.index' are combined as data frames or a vector, respectively. If 'FALSE', these values are returned as lists with entries corresponding to the input objects.

Details

Matching by the estimated propensity score creates matching sets in which treated and untreated observations have identical or similar estimated propensity score. One or more untreated observations will be matched to each treated observation or vice versa.

The caliper, i.e. the maximum distance between the estimated propensity scores of treated and untreated observations to be matched is generally defined as `x`=0.2 of the standard deviation of the (`caliper`)=logit of the estimated propensity score.

If function `pscore()` is previously used with default settings, `matched.by` has not to be specified. It is needed, if the matching variable in data is not labeled by 'pscore'. Also `treat` has not to be specified, contrary to the case where one or two data frames are given as input objects.

Value

`ps.match()` returns an object of class 'matched.pscore', 'matched.data.frame' or 'matched.data.frames' depending on the class(es) of the input object(s) and `combine.output`. If the class of the input object is 'pscore', the output object inherits all components from the input object. The following components are available:

data	a data frame containing the input data, extended by column(s) including the matching indices labeled by name.match.index. If the input object is a data frame, 'data' are sorted by treatment. If object.control is given and combine.output='FALSE', 'data' is a list of two data frames corresponding to the input objects.
data.matched	a data frame limiting 'data' only to matched observations.
matched.by	a string indicating the name of the matching variable.
name.match.index	a string indicating the name of matching indices generated at last.
match.index	a numeric vector containing the matching indices labeled by 'name.match.index' whereas '0' indicates 'no matching partner found'. If combine.output='FALSE', it is a list of two vectors corresponding to the input objects.
match.parameters	a list of matching parameters (caliper, ratio, who.treated, givenTmatchingC, bestmatch.first).
formula.pscore	a formula describing formally the propensity score model fitted at last in pscore().
model.pscore	an object of class glm containing information about the propensity score model fitted at last in pscore().
name.pscore	a string indicating the name of propensity score estimated at last in pscore().
pscore	a numeric vector containing the estimated propensity score labeled by name.pscore.
name.treat	a string indicating the name of the selected treatment variable.
treat	a numeric vector containing the treatment index labeled by name.treat.

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

Examples

```
## STU1
data(stu1)
stu1.ps <- pscore(data    = stu1,
                  formula = therapie~tgr+age)
stu1.match <- ps.match(object          = stu1.ps,
                       ratio           = 2,
                       caliper         = 0.5,
                       givenTmatchingC = FALSE,
                       matched.by      = "pscore",
                       setseed         = 38902)
```

pscore *Propensity score estimation*

Description

Estimate the propensity score using a logistic regression model

Usage

```
pscore(formula, data, family="binomial", na.action=na.exclude,
name.pscore="pscore", ...)
```

Arguments

formula	an object of class 'formula' (or one that can be coerced to that class): a symbolic description of a model to be fitted. The outcome given in formula must be labeled with '0' and '1' due to the internal use of glm.
data	a data frame containing outcome and treatment variable and covariates.
family	the error distribution and link function to be used in the model (see glm). The default is 'binomial'.
na.action	a function which indicates what should happen when data contain 'NA's. The default is 'na.exclude', i.e., data containing 'NA' values are deleted (see na.exclude).
name.pscore	a string indicating the name of the estimated propensity score.
...	further arguments passed to or from other methods.

Details

The propensity score is the conditional probability of receiving a certain treatment given patient's covariates. It is generally unknown and has to be estimated, e.g. by using logistic regression. pscore can be used repeatedly and all estimated propensity scores are added on 'data'. But only the information of the propensity score estimated at last will be stored in values of the output object.

Value

pscore returns an object of class 'pscore' containing the following components:

data	a data frame containing the input data, extended by column(s) including the estimated propensity score(s) labeled by name.pscore.
formula.pscore	a formula describing formally the propensity score model fitted at last.
model.pscore	an object of class glm containing information about the propensity score model fitted at last.
name.pscore	a string indicating the name of the propensity score estimated at last.
pscore	a numeric vector containing the estimated propensity score fitted at last and labeled by 'name.pscore'.

name.treat	a string indicating the name of the treatment variable given in formula as outcome.
treat	a numeric vector containing the treatment labeled by 'name.treat'.

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

See Also

glm, formula

Examples

```
## STU1
data(stu1)
stu1.ps <- pscore(data    = stu1,
                  formula = therapie~tgr+age)


## PRIDE
data(pride)
pride.ps <- pscore(data        = pride,
                   formula     = PCR_RSV~SEX+RSVINF+REGION+
                                 AGE+ELTATOP+EINZ+EXT,
                   name.pscore = "ps")
```

---

relative.effect *Relative effects of covariates*

Description

Estimate the extent to which a covariate is confounding the treatment effect

Usage

```
relative.effect(formula=NULL, data, sel=NULL, resp=NULL, treat=NULL, ...)
```

Arguments

formula	an object of class 'formula' (or one that can be coerced to that class): a symbolic description of a model to be fitted.
data	a data frame containing outcome, treatment and covariates.
sel	a vector of integers or strings indicating the covariates.
resp	an integer or a string indicating the outcome variable.
treat	an integer or a string indicating the treatment variable.
...	further arguments passed to or from other methods.

Details

The decision about the inclusion of covariates in the propensity score model is mostly difficult. A measure describing the extent to which a covariate is confounding the treatment effect on outcome can help to decide on it. Covariates with a large impact are potential candidates for the propensity score model.

The relative effect is defined as difference between adjusted and unadjusted treatment effect related to the unadjusted effect (per cent). Therefore, treatment effects on outcome, unadjusted and adjusted for covariates, are estimated using internally `glm`.

Two options are available to fit appropriate regression models. Either a formula is specified, typically as 'resp ~ treat + cov' (`formula`), or `resp`, `treat` and `sel` are given to specify the outcome and treatment variable and the covariates.

Value

`relative.effect` returns a list containing the following components:

`unadj.treat`	the estimated unadjusted treatment effect on outcome.
`adj.treat.cov`	a vector containing the estimated treatment effects on outcome, individually adjusted for the selected covariates.
`rel.treat`	a vector containing the relative effect for each covariate.
`name.treat`	a string indicating the name of the treatment.
`name.resp`	a string indicating the name of the outcome.
`name.sel`	a vector of strings indicating the names of the selected covariates.
`family`	the error distribution and link function used in the model (see `glm`).

Author(s)

Susanne Stampf <susanne.stampf@usb.ch>

See Also

`glm`, formula

Examples

```
## STU1
data(stu1)
stu1.effect <-
  relative.effect(data = stu1,
                  formula = pst~therapie+tgr+age)

## PRIDE
data(pride)
pride.effect <-
  relative.effect(data = pride,
                  sel = c(2:14),
                  resp = 15,
                  treat = 1)
```

stu1 *Data example stu1*

Description

Data on quality of life in n=646 breast cancer patients in an observational treatment study of the German Breast Cancer Study Group (GBSG)

Usage

```
data(stu1)
```

Format

A data frame with 648 observations on the following 9 variables.

klinik clinic center

idnr patient id

tmass tumor size in mm

alter age

therapie therapy:

- **1:** mastectomy
- **2:** breast conservation

tgr tumor size:

- **1:** <= 10 mm
- **2:** > 10 mm

age age:

- **1:** <= 55 yrs
- **2:** > 55 yrs

ewb emotional status

pst physical status

Examples

```
data(stu1)
```

summary.relative.effect *Summary function*

Description

to print summarized results.

Usage

```
 ## S3 method for class 'relative.effect'
summary(object, ...)
```

Arguments

object	an object to be printed.
...	further arguments passed to.

Details

Summary function

Index