Package 'TriMatch'

December 8, 2025

Type Package

Title Propensity Score Matching of Non-Binary Treatments

Version 1.0.1

Date 2025-12-08

Description Propensity score matching for non-binary treatments.

License GPL (>= 2)

URL https://jbryer.github.io/TriMatch/,
https://github.com/jbryer/TriMatch/

BugReports https://github.com/jbryer/TriMatch/issues/

Depends ggplot2, R (>= 3.0), reshape2, scales

Imports car, compiler, grid, gridExtra, plyr, PSAgraphics, psych, randomForest, stats, stringr

Suggests bookdown, knitr, MASS, rmarkdown, xtable

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Jason Bryer [aut, cre] (ORCID: <https://orcid.org/0000-0002-2454-0402>)

Maintainer Jason Bryer <jason@bryer.org>

Repository CRAN

Date/Publication 2025-12-08 06:30:17 UTC

Contents

`TriMatch-package` *Propensity Score Analysis for Non-Binary Treatments*

Description

This packages provides functions to estimate and visualize propensity score analyses including matching for non-binary treatments.

Author(s)

Jason Bryer <jason@bryer.org>

See Also

PSAgraphics multilevelPSA

---

as.data.frame.list *Convert a list of vectors to a data frame.*

Description

This function will convert a list of vectors to a data frame. This function will handle three different types of lists of vectors. First, if all the elements in the list are named vectors, the resulting data frame will have have a number of columns equal to the number of unique names across all vectors. In cases where some vectors do not have names in other vectors, those values will be filled with NA.

Usage

```
## S3 method for class 'list'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	a list to convert to a data frame.
row.names	a vector equal to length(x) corresponding to the row names. If NULL, the row names will be set to names(x).
optional	not used.
...	other parameters passed to [base::data.frame()].

Details

The second case is when all the vectors are of the same length. In this case, the resulting data frame is equivalent to applying rbind across all elements.

The third case handled is when there are varying vector lengths and not all the vectors are named. This condition should be avoided. However, the function will attempt to convert this list to a data frame. The resulting data frame will have a number of columns equal to the length of the longest vector. For vectors with length less than this will fill the row with NAs. Note that this function will print a warning if this condition occurs.

Value

a data frame.

Author(s)

Jason Bryer jason@bryer.org

References

http://stackoverflow.com/questions/4227223/r-list-to-data-frame

Examples

```
    test1 <- list( c(a='a',b='b',c='c'), c(a='d',b='e',c='f'))
    as.data.frame(test1)

    test2 <- list( c('a','b','c'), c(a='d',b='e',c='f'))
    as.data.frame(test2)

    test3 <- list('Row1'=c(a='a',b='b',c='c'), 'Row2'=c(var1='d',var2='e',var3='f'))
    as.data.frame(test3)

    ## Not run:
    #This will print a warning.
    test4 <- list('Row1'=letters[1:5], 'Row2'=letters[1:7], 'Row3'=letters[8:14])
    as.data.frame(test4)

## End(Not run)

    test5 <- list(letters[1:10], letters[11:20])
    as.data.frame(test5)

    ## Not run:
    #This will throw an error.
    test6 <- list(list(letters), letters)
    as.data.frame(test6)

## End(Not run)
```

`balance.plot` *Balance plot for the given covariate.*

Description

If the covariate is numeric, boxplots will be drawn with red points for the mean and green error bars for the standard error. For non-numeric covariates a barplot will be drawn.

Usage

```
balance.plot(
  x,
  covar,
  model,
  nstrata = attr(attr(tmatch, "triangle.psa"), "nstrata"),
  label = "Covariate",
  ylab = "",
  xlab = NULL,
  se.ratio = 2,
  print = TRUE,
  legend.position = "top",
```

```
  x.axis.labels,
  x.axis.angle = -45,
  ...
)
```

Arguments

x	results from [trimatch()].
covar	vector of the covariate to check balance of.
model	an integer between 1 and 3 indicating from which model the propensity scores will be used.
nstrata	number of strata to use.
label	label for the legend.
ylab	label of the y-axis.
xlab	label of the x-axis.
se.ratio	a multiplier for how large standard error bars will be.
print	print the output if the Freidman Rank Sum Test and repeated measures ANOVA (for continuous variables).
legend.position	the position of the legend.
x.axis.labels	labels for the x-axis.
x.axis.angle	angle for x-axis labels.
...	parameters passed to [plot.balance.plots()].

Details

A Friedman rank sum test will be performed for all covariate types, printed, and stored as an attribute to the returned object named `friedman`. If a continuous covariate a repeated measures ANOVA will also be performed, printed, and returned as an attribute named `rmanova`.

Value

a `ggplot2` figure or a list of `ggplot2` figures if `covar` is a data frame.

---

boxdiff.plot *Returns a* ggplot2 *box plot of the differences.*

Description

A boxplot of differences between each pair of treatments.

Usage

```
boxdiff.plot(
  tmatch,
  out,
  plot.mean = TRUE,
  ordering = attr(tmatch, "match.order"),
  ci.width = 0.5
)
```

Arguments

tmatch	the results from [trimatch()].
out	a vector of the outcome measure of interest.
plot.mean	logical indicating whether the means should be plotted.
ordering	specify the order for doing the paired analysis, that is analysis will be conducted as: ordering[1] - ordering[2], ordering[1] - ordering[3], and ordering[2] - ordering[3].
ci.width	the width for the confidence intervals.

Value

a ggplot2 boxplot of the differences.

---

covariateBalance *Calculate covariate effect size differences before and after stratification.*

Description

This function is modified from the [PSAgraphics::cv.bal.psa()] function in the 'PSAgrpahics' package.

Usage

```
covariateBalance(
  covariates,
  treatment,
  propensity,
  strata = NULL,
  int = NULL,
  tree = FALSE,
  minsize = 2,
  universal.psd = TRUE,
  trM = 0,
  absolute.es = TRUE,
  trt.value = NULL,
```

```
  use.trt.var = FALSE,
  verbose = FALSE,
  xlim = NULL,
  plot.strata = TRUE,
  ...
)
```

Arguments

covariates	dataframe of interest
treatment	binary vector of 0s and 1s (necessarily? what if character, or 1, 2?)
propensity	PS scores from some method or other.
strata	either a vector of strata number for each row of covariate, or one number n in which case it is attempted to group rows by ps scores into n strata of size approximately 1/n. This does not seem to work well in the case of few specific propensity values, as from a tree.
int	either a number m used to divide [0,1] into m equal length subintervals, or a vector of cut points between 0 an 1 defining the subintervals (perhaps as suggested by loess.psa). In either case these subintervals define strata, so strata can be of any size.
tree	logical, if unique ps scores are few, as from a recursively partitioned tree, then TRUE will force each ps value to define a stratum.
minsize	smallest allowable stratum-treatment size. If violated, strata is removed.
universal.psd	If 'TRUE', forces standard deviations used to be unadjusted for stratification.
trM	trimming proportion for mean calculations.
absolute.es	logical, if 'TRUE' routine uses absolute values of all effect sizes.
trt.value	allows user to specify which value is active treatment, if desired.
use.trt.var	logical, if true then Rubin-Stuart method using only treatment variance with be used in effect size calculations.
verbose	logical, controls output that is visibly returned.
xlim	limits for the x-axis.
plot.strata	logical indicating whether to print strata.
...	currently unused.

Details

Note: effect sizes are calculated as treatment 1 - treatment 0, or treatment B - treatment A.

Author(s)

Robert M. Pruzek RMPruzek@yahoo.com

James E. Helmreich James.Helmreich@Marist.edu

KuangNan Xiong harryxkn@yahoo.com

`data.frame.to.list` *Convert a list of vectors to a data frame.*

Description

This function will convert a list of vectors to a data frame. This function will handle three different types of lists of vectors. First, if all the elements in the list are named vectors, the resulting data frame will have have a number of columns equal to the number of unique names across all vectors. In cases where some vectors do not have names in other vectors, those values will be filled with `NA`.

Usage

```
data.frame.to.list(...)
```

Arguments

`...`	other parameters passed to [base::data.frame()].

Details

The second case is when all the vectors are of the same length. In this case, the resulting data frame is equivalent to applying `rbind` across all elements.

The third case handled is when there are varying vector lengths and not all the vectors are named. This condition should be avoided. However, the function will attempt to convert this list to a data frame. The resulting data frame will have a number of columns equal to the length of the longest vector. For vectors with length less than this will fill the row with `NA`s. Note that this function will print a warning if this condition occurs.

Value

a data frame.

Author(s)

Jason Bryer jason@bryer.org

References

http://stackoverflow.com/questions/4227223/r-list-to-data-frame

Examples

```
test1 <- list( c(a='a',b='b',c='c'), c(a='d',b='e',c='f'))
as.data.frame(test1)

test2 <- list( c('a','b','c'), c(a='d',b='e',c='f'))
as.data.frame(test2)
```

```
    test3 <- list('Row1'=c(a='a',b='b',c='c'), 'Row2'=c(var1='d',var2='e',var3='f'))
    as.data.frame(test3)

    ## Not run:
    #This will print a warning.
    test4 <- list('Row1'=letters[1:5], 'Row2'=letters[1:7], 'Row3'=letters[8:14])
    as.data.frame(test4)

## End(Not run)

    test5 <- list(letters[1:10], letters[11:20])
    as.data.frame(test5)

    ## Not run:
    #This will throw an error.
    test6 <- list(list(letters), letters)
    as.data.frame(test6)

## End(Not run)
```

`distance.euclid` *Euclidean distance calculation.*

Description

This method uses a simple Euclidean distance calculation for determining the distances between two matches. That is, |ps1 - ps2|.

Usage

```
distance.euclid(x, grouping, id, groups, caliper, nmatch = Inf)
```

Arguments

`x`	vector of propensity scores.
`grouping`	vector or factor identifying group membership.
`id`	vector corresponding to unique identifer for each element in `x` and `grouping`.
`groups`	vector of length two indicating the unique groups to calculate the distance between. The first element will be the rows, the second columns.
`caliper`	a scaler indicating the caliper to use for matching within each step.
`nmatch`	number of smallest distances to retain.

Value

a list of length equal to `x`. Each element of the list is a named numeric vector where the values correspond to the distance and the name to the `id`.

`distances.plot` *Barplot for the sum of distances.*

Description

Barplot for the sum of distances.

Usage

```
distances.plot(tmatch, caliper = 0.25, label = FALSE)
```

Arguments

`tmatch`	the results of [trimatch()].
`caliper`	a vector indicating where vertical lines should be drawn as a factor of the standard deviation. Rosenbaum and Rubin (1985) suggested one quarter of one standard deviation.
`label`	label the bars that exceed the minimum caliper.

See Also

triangle.match

`ezANOVA` *Compute ANOVA*

Description

Note that this adapted from the 'ez' R package available here: https://github.com/mike-lawrence/ez It appears the 'ez' package is no longer maintained so the function was copied here.

Usage

```
ezANOVA(
  data,
  dv,
  wid,
  within = NULL,
  within_full = NULL,
  within_covariates = NULL,
  between = NULL,
  between_covariates = NULL,
  observed = NULL,
  diff = NULL,
  reverse_diff = FALSE,
```

```
  type = 1,
  white.adjust = FALSE,
  detailed = FALSE,
  return_aov = FALSE
)
```

Arguments

data	Data frame containing the data to be analyzed.
dv	Name of the column in data that contains the dependent variable. Values in this column must be numeric.
wid	Name of the column in data that contains the variable specifying the case/Ss identifier. This should be a unique value per case/Ss.
within	Names of columns in data that contain predictor variables that are manipulated (or observed) within-Ss. If a single value, may be specified by name alone; if multiple values, must be specified as a .() list.
within_full	Same as within, but intended to specify the full within-Ss design in cases where the data have not already been collapsed to means per condition specified by within and when within only specifies a subset of the full design.
within_covariates	Names of columns in data that contain predictor variables that are manipulated (or observed) within-Ss and are to serve as covariates in the analysis. If a single value, may be specified by name alone; if multiple values, must be specified as a .() list.
between	Names of columns in data that contain predictor variables that are manipulated (or observed) between-Ss. If a single value, may be specified by name alone; if multiple values, must be specified as a .() list.
between_covariates	Names of columns in data that contain predictor variables that are manipulated (or observed) between-Ss and are to serve as covariates in the analysis. If a single value, may be specified by name alone; if multiple values, must be specified as a .() list.
observed	Names of columns in data that are already specified in either within or between that contain predictor variables that are observed variables (i.e. not manipulated). If a single value, may be specified by name alone; if multiple values, must be specified as a .() list. The presence of observed variables affects the computation of the generalized eta-squared measure of effect size reported by ezANOVA.
diff	Names of any variables to collapse to a difference score. If a single value, may be specified by name alone; if multiple values, must be specified as a .() list. All supplied variables must be factors, ideally with only two levels (especially if setting the reverse_diff argument to TRUE).
reverse_diff	Logical. If TRUE, triggers reversal of the difference collapse requested by diff. Take care with variables with more than 2 levels.
type	Numeric value (either 1, 2 or 3) specifying the Sums of Squares "type" to employ when data are unbalanced (eg. when group sizes differ). type = 2 is the

default because this will yield identical ANOVA results as type = 1 when data are balanced but type = 2 will additionally yield various assumption tests where appropriate. When data are unbalanced, users are warned that they should give special consideration to the value of type. type=3 will emulate the approach taken by popular commercial statistics packages like SAS and SPSS, but users are warned that this approach is not without criticism.

`white.adjust` Only affects behavior if the design contains only between-Ss predictor variables. If not FALSE, the value is passed as the white.adjust argument to Anova, which provides heteroscedasticity correction. See Anova for details on possible values.

`detailed` Logical. If TRUE, returns extra information (sums of squares columns, intercept row, etc.) in the ANOVA table.

`return_aov` Logical. If TRUE, computes and returns an aov object corresponding to the requested ANOVA (useful for computing post-hoc contrasts).

Details

This function provides easy analysis of data from factorial experiments, including purely within-Ss designs (a.k.a. "repeated measures"), purely between-Ss designs, and mixed within-and-between-Ss designs, yielding ANOVA results, generalized effect sizes and assumption checks.

ANCOVA is implemented by first regressing the DV against each covariate (after collapsing the data to the means of that covariate's levels per subject) and subtracting from the raw data the fitted values from this regression (then adding back the mean to maintain scale). These regressions are computed across Ss in the case of between-Ss covariates and computed within each Ss in the case of within-Ss covariates.

Warning

Prior to running (though after obtaining running ANCOVA regressions as described in the details section), dv is collapsed to a mean for each cell defined by the combination of wid and any variables supplied to within and/or between and/or diff. Users are warned that while convenient when used properly, this automatic collapsing can lead to inconsistencies if the pre-collapsed data are unbalanced (with respect to cells in the full design) and only the partial design is supplied to ezANOVA. When this is the case, use within_full to specify the full design to ensure proper automatic collapsing.

Value

A list containing one or more of the following components:

ANOVA A data frame containing the ANOVA results.

Mauchly's Test for Sphericity If any within-Ss variables with >2 levels are present, a data frame containing the results of Mauchly's test for Sphericity. Only reported for effects >2 levels because sphericity necessarily holds for effects with only 2 levels.

Sphericity Corrections If any within-Ss variables are present, a data frame containing the Greenhouse-Geisser and Huynh-Feldt epsilon values, and corresponding corrected p-values.

Levene's Test for Homogeneity If the design is purely between-Ss, a data frame containing the results of Levene's test for Homogeneity of variance. Note that Huynh-Feldt corrected p-values where the Huynh-Feldt epsilon >1 will use 1 as the correction epsilon.

aov An aov object corresponding to the requested ANOVA.

Some column names in the output data frames are abbreviated to conserve space:

DFn Degrees of Freedom in the numerator (a.k.a. DFeffect).

DFd Degrees of Freedom in the denominator (a.k.a. DFerror).

SSn Sum of Squares in the numerator (a.k.a. SSeffect).

SSd Sum of Squares in the denominator (a.k.a. SSerror).

F F-value.

p p-value (probability of the data given the null hypothesis).

p<.05 Highlights p-values less than the traditional alpha level of .05.

ges Generalized Eta-Squared measure of effect size (see in references below: Bakeman, 2005).

GGe Greenhouse-Geisser epsilon.

p[GGe] p-value after correction using Greenhouse-Geisser epsilon.

p[GGe <.05] Highlights p-values (after correction using Greenhouse-Geisser epsilon) less than the traditional alpha level of .05.

HFe Huynh-Feldt epsilon.

p[HFe] p-value after correction using Huynh-Feldt epsilon.

p[HFe <.05] Highlights p-values (after correction using Huynh-Feldt epsilon) less than the traditional alpha level of .05.

W Mauchly's W statistic

Author(s)

Mike Lawrence

References

Bakeman, R. (2005). Recommended effect size statistics for repeated measures designs. Behavior Research Methods, 37 (3), 379-384.

`loess3.plot` *Loess plot for matched triplets.*

Description

This function will create a `ggplot2` figure with propensity scores on the x-axis and the outcome on the y-axis. Three Loess regression lines will be plotted based upon the propensity scores from `model`. Since each model produces propensity scores for two of the three groups, the propensity score for the third group in each matched triplet will be the mean of the other two. If `model` is not specified, the default will be to use the model that estimates the propensity scores for the first two groups in the matching order.

Usage

```
loess3.plot(
  tmatch,
  outcome,
  model,
  ylab = "Outcome",
  plot.connections = FALSE,
  connections.color = "black",
  connections.alpha = 0.2,
  plot.points = geom_point,
  points.alpha = 0.1,
  points.palette = "Dark2",
  ...
)
```

Arguments

`tmatch`	the results of [trimatch()].
`outcome`	a vector representing the outcomes.
`model`	an integer between 1 and 3 indicating from which model the propensity scores will be used.
`ylab`	the label for the y-axis.
`plot.connections`	boolean indicating whether lines will be drawn connecting each matched triplet.
`connections.color`	the line color of connections.
`connections.alpha`	number between 0 and 1 representing the alpha levels for connection lines.
`plot.points`	a `ggplot2` function for plotting points. Usually [ggplot2::geom_point()] or [ggplot2::geom_jitter()]. If 'NULL' no points will be drawn.
`points.alpha`	number between 0 and 1 representing the alpha level for the points.
`points.palette`	the color palette to use. See [ggplot2::scale_colour_brewer()] and https://colorbrewer2.org for more information.
`...`	other parameters passed to [ggplot2::geom_smooth()] and [ggplot2::stat_smooth()].

Value

a 'ggplot2' figure.

maximumTreat *This method will return at least one treatment from groups one and two within the caliper.*

Description

This method will attempt to return enough rows to use each treatment (the first two groups in the matching order) at least once. Assuming treat1 is the first group in the match order and treat2 the second, all duplicate treat1 rows are removed. Next, all treat2 units not in present in after removing duplicate treat1 units are identified. For each of those treat2 units, the matched triplet with the smallest overall distances where treat2 is one of the mathched units is retained.

Usage

```
maximumTreat(tmatch, ...)
```

Arguments

tmatch	initial results from [trimatch()] that contains all possible matches within the specified caliper.
...	currently unused.

merge.triangle.matches *Merges outcomes with the matched set.*

Description

The y parameter should be a subset of the original data used.

Usage

```
## S3 method for class 'triangle.matches'
merge(x, y, ...)
```

Arguments

x	the result of [trimatch()]
y	another data frame or vector to merge with.
...	unused

Value

x with the additional column(s) added.

merge.triangle.psa *Merges covariate(s) with the results of [trips()].*

Description

The y parameter should be a subset of the original data used.

Usage

```
## S3 method for class 'triangle.psa'
merge(x, y, ...)
```

Arguments

x	the result of [trips()]
y	another data frame or vector to merge with.
...	unused

Value

x with the additional column(s) added.

multibalance.plot *Multiple covariate balance assessment plot.*

Description

A graphic based upon [cv.bal.psa()] function in the 'PSAgraphics' package. This graphic plots the effect sizes for multiple covariates before and after propensity score adjustment.

Usage

```
multibalance.plot(tpsa, tmatch, grid = TRUE, cols)
```

Arguments

tpsa	results of [trips()].
tmatch	results of [trimatch()].
grid	if TRUE, then a grid of three plots for each model will be displayed.
cols	character vector of covariates (i.e. column names) from the original data to include in the plot. By default all covariates used in the logistic regression model are used.

Value

a ggplot2 figure.

nmes — *Results from the 1987 National Medical Expenditure Study*

Description

This file was originally prepared by Anders Corr (corr@fas.harvard.edu) who reports on December 8, 2007 that the resulting numbers closely match with those reported in the published article. It was later modified by Jason Bryer (jason@bryer.org) to an R data object to be included in this package. See http://imai.princeton.edu/research/pscore.html for more information

Format

a data frame with 9,708 observations of 12 variables.

Author(s)

United States Department of Health and Human Services. Agency for Health Care Policy and Research

Source

http://imai.princeton.edu/research/pscore.html

References

National Center For Health Services Research, 1987. National Medical Expenditure Survey. Methods II. Questionnaires and data collection methods for the household survey and the Survey of American Indians and Alaska Natives. National Center for Health Services Research and Health Technology Assessment.

Imai, K., & van Dyk, D.A. (2004). Causal Inference With General Treatment Regimes: Generalizing the Propensity Score, Journal of the American Statistical Association, 99(467), pp. 854-866.

Elizabeth Johnson, E., Dominici, F., Griswold, M., & Zeger, S.L. (2003). Disease cases and their medical costs attributable to smoking: An analysis of the national medical expenditure survey. Journal of Econometrics, 112.

OneToN — *This method will use a M1-to-M2-to-1 matching.*

Description

In this method, `M2` corresponds to the number of times a treat1 unit can be matched with a treat2 unit. The `M1` parameter corresponds to the number of times a treat1 unit can be used in total.

Usage

```
OneToN(tmatch, M1 = 2, M2 = 1, ...)
```

Arguments

tmatch	initial results from [trimatch()] that contains all possible matches within the specified caliper.
M1	a scaler indicating the number of unique subjects in group one to retain. This applies only to the first group in the matching order.
M2	a scaler indicating the number of unique matches to retain. This applies to the first two groups in the matching order.
...	currently unused.

---

parallel.plot *Parallel coordinate plot for the three groups and dependent variable.*

Description

Creates a ggplot2 figure of a parallel coordinate plot.

Usage

```
parallel.plot(tmatch, outcome)
```

Arguments

tmatch	results from [trimatch()].
outcome	vector of the outcome

---

perpPt *Internal method for plotting. Finds a point d distance from x, y*

Description

Internal method for plotting. Finds a point d distance from x, y

Usage

```
perpPt(x, y, d = 0.05)
```

Arguments

x	x coordinate
y	y coordinate
d	the distance

plot.balance.plots *Prints a grid of balance plots.*

Description

Prints a grid of balance plots.

Usage

```
## S3 method for class 'balance.plots'
plot(x, rows, cols, byrow = TRUE, plot.sequence = seq_along(bplots), ...)
```

Arguments

x	the results of [balance.plot()] when a data frame is specified.
rows	if covar is a data frame of covariates, the number of rows in the grid of figures.
cols	if covar is a data frame of covariates, the number of columns in the grid of figures.
byrow	if TRUE (default), plots will be drawn by rows, otherwise by columns.
plot.sequence	the sequence (or subset) of plots to draw.
...	currently unused.

plot.triangle.matches *Triangle plot drawing matched triplets.*

Description

This plot function adds a layer to [plot.triangle.psa()] drawing matched triplets. If ‘p‘ is supplied, this function will simply draw on top of the pre-existing plot, otherwise [plot.triangle.psa()] will be called first.

Usage

```
## S3 method for class 'triangle.matches'
plot(
  x,
  sample = 0.05,
  rows = sample(nrow(tmatch), nrow(tmatch) * sample),
  line.color = "black",
  line.alpha = 0.5,
  point.color = "black",
  point.size = 3,
  p,
  ...
)
```

Arguments

x	matched triplets from [triangle.match()].
sample	an number between 0 and 1 representing the percentage of matched triplets to draw.
rows	an integer vector corresponding to the rows in ‘tmatch‘ to draw.
line.color	the line color.
line.alpha	the alpha for the lines.
point.color	color of matched triplet points.
point.size	point size for matched triplets.
p	a ggplot to add the match lines. If NULL, then [plot.triangle.psa()].
...	other parameters passed to [plot.triangle.psa()].

Details

If this function calls [plot.triangle.psa()], it will only draw line segments and points for those data rows that were used in the matching procedure. That is, data elements not matched will be excluded from the figure. To plot all segments and points regardless if used in matching, set ‘p = plot(tpsa)‘.

Value

a ‘ggplot2‘ graphic.

See Also

plot.triangle.psa

triangle.match

---

plot.triangle.psa *Triangle plot.*

Description

Triangle plot showing the fitted values (propensity scores) for three different models.

Usage

```
## S3 method for class 'triangle.psa'
plot(
  x,
  point.alpha = 0.3,
  point.size = 1.5,
  legend.title = "Treatment",
  text.size = 4,
  draw.edges = FALSE,
```

```
  draw.segments = TRUE,
  edge.alpha = 0.2,
  edge.color = "grey",
  edge.labels = c("Model 1", "Model 2", "Model 3"),
  sample = c(1),
  ...
)
```

Arguments

x	the results from [trips()].
point.alpha	alpha level for points.
point.size	point size.
legend.title	title for the legend.
text.size	text size.
draw.edges	draw edges of the triangle.
draw.segments	draw segments connecting points across two models.
edge.alpha	alpha level for edges if drawn.
edge.color	the color for edges if drawn.
edge.labels	the labels to use for each edge of the triangle.
sample	a vector of length 1 or 3 representing the sample of points to plot. The position of each element corresponds to the groups as returned by `attr(tpsa,'groups')`. If equal to one, all points will be plotted. Values less than one will plot a percentage of points. Values greater than one exactly that number of points will be plotted.
...	currently unused.

Value

ggplot2 figure

See Also

triangle.psa

---

print.balance.plots *Print the results of [balance.plot()] for a data frame of covariates.*

Description

Print the results of [balance.plot()] for a data frame of covariates.

Usage

```
## S3 method for class 'balance.plots'
print(x, ...)
```

Arguments

x	the results of [balance.plot()] when a data frame is specified.
...	parameters passed to [plot.balance.plots()] and [summary.balance.plots()].

---

print.triangle.plot *Print method for [plot.triangle.psa()]. The primary purpose is to suppress the "Removed n rows containing missing values" warning printed by 'ggplot2'.*

Description

Print method for [plot.triangle.psa()]. The primary purpose is to suppress the "Removed n rows containing missing values" warning printed by 'ggplot2'.

Usage

```
## S3 method for class 'triangle.plot'
print(x, ...)
```

Arguments

x	a plot from [plot.triangle.psa()].
...	other parameters passed to ggplot2.

---

print.trimatch.summary *Prints the results of [summary.triangle.matches()].*

Description

This is an S3 generic function to print the results of [summary.triangle.matches()].

Usage

```
## S3 method for class 'trimatch.summary'
print(x, ...)
```

Arguments

x	results of [summary.triangle.matches()].
...	multiple results of [summary.triangle.matches()]. These must be named. For example, `"Method 1" = summary(tmath, outcome)`.

segment1 *Internal method for plotting. Position along the left side segment*

Description

Internal method for plotting. Position along the left side segment

Usage

```
segment1(d)
```

Arguments

d	the distance

segment2 *Internal method for plotting. Position along the right side segment*

Description

Internal method for plotting. Position along the right side segment

Usage

```
segment2(d)
```

Arguments

d	the distance

star *Returns significance level.*

Description

Returns the significance level as stars, or NA if a non-numeric value is passed in.

Usage

```
star(x)
```

Arguments

x	p-value.

summary.balance.plots *Prints a summary table of the test statistics of each balance plot.*

Description

The [balance.plot()] function will create a grid of balance plots if a data frame is provided. The returned object is a list of ggplot2 figures with the statistical tests (i.e. Friedmen Rank Sum tests and if a continuous variable, repeated measures ANOVA as well) saved as attributes. This function will return a data frame combining all of those results.

Usage

```
## S3 method for class 'balance.plots'
summary(object, ...)
```

Arguments

object	the results of [balance.plot()] when a data frame is specified.
...	currently unused.

Value

a data frame

summary.triangle.matches *Provides a summary of the matched triplets including analysis of outcome measure if provided.*

Description

If an outcome measure is provided this function will perform a Freidman Rank Sum Test and repeated measures ANOVA. If either test has a statistically significant difference (as determined by the value of the p parameter), a Pairwise Wilcoxon Rank Sum Test will also be provided.

Usage

```
## S3 method for class 'triangle.matches'
summary(object, outcome, p = 0.05, ordering = attr(object, "match.order"), ...)
```

Arguments

object	result of [trimatch()].
outcome	vector representing the outcome measure.
p	threshold of the p value to perform a
ordering	specify the order for doing the paired analysis, that is analysis will be conducted as: ordering[1] - ordering[2], ordering[1] - ordering[3], and ordering[2] - ordering[3].
...	parameters passed to other statistical tests.

Value

a trimatch.summary object.

See Also

[stats::friedman.test()], ez::ezANOVA(), [stats::pairwise.wilcox.test()]

---

summary.triangle.psa *Prints the summary results of the logistic regression models.*

Description

The [trips()] function estimates three separate logistic regression models for each pair of groups. This function will print a combined table of the three summaries.

Usage

```
## S3 method for class 'triangle.psa'
summary(object, ...)
```

Arguments

object	the results of [trips()].
...	currently unused.

summary.unmatched *Provides a summary of unmatched subjects.*

Description

Will return as a list and print the percentage of total unmatched rows and percent by treatment.

Usage

```
## S3 method for class 'unmatched'
summary(object, digits = 3, ...)
```

Arguments

object	results of [unmatched()].
digits	number of digits to print.
...	currently unused.

Value

a list of summary results.

trimatch *Creates matched triplets.*

Description

Create matched triplets by minimizing the total distance between matched triplets within a specified caliper.

Usage

```
trimatch(
  tpsa,
  caliper = 0.25,
  nmatch = c(15),
  match.order,
  exact,
  method = maximumTreat,
  ...
)
```

Arguments

tpsa	the results from [trips()]
caliper	a vector of length one or three indicating the caliper to use for matching within each step. This is expressed in standardized units such that .25 means that matches must be within .25 of one standard deviation to be kept, otherwise the match is dropped.
nmatch	number of closest matches to retain before moving to next edge. This can be Inf in which case all matches within the caliper will be retained through to the next step. For large datasets, evaluating all possible matches within the caliper could be time consuming.
match.order	character vector of length three indicating the order in which the matching algorithm will processes. The default is to use start with the group the middle number of subjects, followed by the smallest, and then the largest.
exact	a vector or data frame of representing covariates for exact matching. That is, matched triplets will first be matched exactly on these covariates before evaluating distances.
method	This is a function that specifies which matched triplets will be retained. If 'NULL', all matched triplets within the specified caliper will be returned (equivalent to caliper matching in two group matching). The default is [maximumTreat()] that attempts include each treatment at least once. Another option is [OneToN()] which mimicks the one-to-n matching where treatments are matched to multiple control units.
...	other parameters passed to 'method'.

Details

The [trips()] function will estimate the propensity scores for three models. This method will then find the best matched triplets based upon minimizing the summed differences between propensity scores across the three models. That is, the algorithm works as follows:

- The first subject from model 1 is selected.
- The nmatch[1] smallest distances are selected using propensity scores from model 1.
- For each of the matches identified, the subjects propensity score from model 2 is retrieved.
- The nmatch[2] smallest distances are selected using propensity score from model 3.
- For each of those matches identified, the subjects propensity score from model 2 is retrieved.
- The distances is calculated from the first and last subjects propensity scores from model 2.
- The three distances are summed.
- The triplet with the smallest overall distance is selected and returned.

Examples

```
## Not run:
data(turoing)
formu <- ~ Gender + Ethnicity + Military + ESL + EdMother + EdFather + Age +
     Employment + Income + Transfer + GPA
```

```
tpsa <- trips(tutoring, tutoring$treat, formu)
tmatch <- trimatch(tpsa, status=FALSE)

## End(Not run)
```

`trimatch.apply2` *Recursive function to find possible matched triplets using the apply functions.*

Description

Internal method. This version does not use the exact matching. Instead, this function should be called separately for each grouping.

Usage

```
trimatch.apply2(tpsa, caliper, nmatch, match.order, sd1, sd2, sd3)
```

Arguments

`tpsa`	the results from [trips()]
`caliper`	a vector of length one or three indicating the caliper to use for matching within each step. This is expressed in standardized units such that .25 means that matches must be within .25 of one standard deviation to be kept, otherwise the match is dropped.
`nmatch`	number of closest matches to retain before moving to next edge. This can be `Inf` in which case all matches within the caliper will be retained through to the next step. For large datasets, evaluating all possible matches within the caliper could be time consuming.
`match.order`	character vector of length three indicating the order in which the matching algorithm will processes. The default is to use start with the group the middle number of subjects, followed by the smallest, and then the largest.
`sd1`	standard deviation for propensity scores from model 1.
`sd2`	standard deviation for propensity scores from model 2.
`sd3`	standard deviation for propensity scores from model 3.

`trips` *Estimates propensity scores for three groups*

Description

The propensity score is

$$e(X) = P(W = 1|X)$$

This function will estimate the propensity scores for each pair of groups (e.g. two treatments and one control).

Usage

```
trips(
  thedata,
  treat,
  formu = ~.,
  groups = unique(treat),
  nstrata = 5,
  method = "logistic",
  ...
)
```

Arguments

`thedata`	the data frame.
`treat`	vector or factor indicating the treatment/control assignment for `thedata`. Length must be equal to `nrow(thedata)`.
`formu`	the logistic regression formula. Note that the dependent variable should not be specified and will be modified.
`groups`	a vector of exactly length three corresponding the values in `treat` for each control/treatment.
`nstrata`	the number of strata marks to plot on the edge.
`method`	the method to use to estimate the propensity scores. Current options are logistic or randomForest.
`...`	other parameters passed to [stats::glm()].

Details

$$PS_1 = e(X_{T_1C}) = Pr(z = 1|X_{T_1C})$$

$$PS_2 = e(X_{T_2C}) = Pr(z = 1|X_{T_2C})$$

$$PS_3 = e(X_{T_2T_1}) = Pr(z = 1|X_{T_2T_1})$$

Examples

```
## Not run:
data(tutoring)
formu <- ~ Gender + Ethnicity + Military + ESL + EdMother + EdFather + Age +
     Employment + Income + Transfer + GPA
tpsa <- trips(tutoring, tutoring$treat, formu)
head(tpsa)

## End(Not run)
```

`tutoring` *Results from a study examining the effects of tutoring services on course grades.*

Description

- `treat` Treatment indicator.
- `Course` The course id the student was enrolled in.
- `Grade` The course grade the student earned (4=A, 3=B, 2=C, 1=D, 0=F or W).
- `Gender` Gender of the student.
- `Ethnicity` Ethnicity of the student, either White, Black, or Other.
- `Military` Is the student an active military student.
- `ESL` English second language student.
- `EdMother` Education level of the mother (1 = did not finish high school; 2 = high school grad; 3 = some college; 4 = earned associate degree; 5 = earned baccalaureate degree; 6 = Earned Master's degree; 7 = earned doctorate).
- `EdFather` Education level of the father (levels same as EdMother).
- `Age` Age at the start of the course.
- `Employment` Employment level at college enrollment (1 = No; 2 = part-time; 3 = full-time).
- `Income` Household income level at college enrollment (1 = <25K; 2 = <35K; 3 = <45K; 4 = <55K; 5 = <70K; 6 = <85K; 7 = <100K; 8 = <120K; 9 = >120K).
- `Transfer` Number of transfer credits at the start of the course.
- `GPA` GPA as of the start of the course.
- `GradeCode` Letter grade.
- `Level` Level of the course, either Lower or Upper.
- `ID` Randomly assigned student ID.

Format

a data frame with 17 variables.

unmatched *Returns rows from [trips()] that were not matched by [trimatch()].*

Description

This function returns a subset of [trips()] that were not matched by [trimatch()]. All data frame methods work with the returned object but special 'summary' function will provided relevant information.

Usage

```
unmatched(tmatch)
```

Arguments

tmatch	the results of [trimatch()].

Value

a data frame of unmatched rows.

Index