Package 'synthpop'

March 20, 2019

Type Package

Title Generating Synthetic Versions of Sensitive Microdata for Statistical Disclosure Control

Version 1.5-1

Date 2019-03-20

Author Beata Nowok, Gillian M Raab, Joshua Snoke and Chris Dibben

Maintainer Beata Nowok <beata.nowok@gmail.com>

Description A tool for producing synthetic versions of microdata containing confidential information so that they are safe to be released to users for exploratory analysis. The key objective of generating synthetic data is to replace sensitive original values with synthetic ones causing minimal distortion of the statistical information contained in the data set. Variables, which can be categorical or continuous, are synthesised one-by-one using sequential modelling. Replacements are generated by drawing from conditional distributions fitted to the original data using parametric or classification and regression trees models. Data are synthesised via the function syn() which can be largely automated, if default settings are used, or with methods defined by the user. Optional parameters can be used to influence the disclosure risk and the analytical quality of the synthesised data. For a description of the implemented method see Nowok, Raab and Dibben (2016) <doi:10.18637/jss.v074.i11>.

License GPL-2 | GPL-3

Depends lattice, MASS, methods, nnet, ggplot2

Imports graphics, stats, utils, rpart, party, foreign, plyr, proto, polspline, randomForest, classInt, mipfp

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2019-03-20 17:00:03 UTC

R topics documented:

`synthpop-package` *Generating synthetic versions of sensitive microdata for statistical disclosure control*

Description

Generate synthetic versions of a data set using parametric or CART methods.

Details

Package:	synthpop
Type:	Package
Version:	1.5-1
Date:	2019-03-20
License:	GPL-2 | GPL-3

Synthetic data are generated from the original (observed) data by the function `syn`. The package includes also tools to compare synthetic data with the observed data (`compare.synds`) and to fit (generalized) linear model to synthetic data (`lm.synds`, `glm.synds`) and compare the estimates with those for the observed data (`compare.fit.synds`). More extensive documentation with illustrative examples is provided in the package vignette.

Author(s)

Beata Nowok, Gillian M Raab, Joshua Snoke and Chris Dibben based on package **mice** (2.18) by Stef van Buuren and Karin Groothuis-Oudshoorn

Maintainer: Beata Nowok <beata.nowok@gmail.com>

References

Nowok, B., Raab, G.M and Dibben, C. (2016). synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software*, **74**(11), 1-26. doi: 10.18637/jss.v074.i11.

`codebook.syn` *Makes a codebook from a data frame*

Description

Describes features of variables in a data frame relevant for synthesis.

Usage

```
codebook.syn(data, maxlevs = 3)
```

Arguments

`data`	a data frame with a data set to be synthesised.
`maxlevs`	the number of factor levels above which separate tables with all labels are returned as part of `labs` component.

Value

A list with two components.

tab - a data frame with the following information about each variable:

name	variable name
class	class of variable
nmiss	number of missing values (NA)
perctmiss	percentage of missing values
ndistinct	number of distinct values (excluding missing values)
details	range for numeric variables, maximum length for character variables, labels for factors with <= maxlevs levels

labs - a list of extra tables with labels for each factor with number of levels greater than maxlevs.

Examples

```
codebook.syn(SD2011)
```

---

compare *Comparison of synthesised and observed data*

Description

A generic function for comparison of synthesised and observed data. The function invokes particular methods which depend on the class of the first argument.

Usage

```
compare(object, data, ...)
```

Arguments

object	a synthetic data object of class synds or fit.synds.
data	an original observed data set.
...	additional arguments specific to a method.

Details

Compare methods facilitate quality assessment of synthetic data by comapring them with the original observed data sets. The data themselves (for class synds) or models fitted to them (for class fit.synds) are compared.

Value

The value returned by compare depends on the class of its argument. See the documentation of the particular methods for details.

See Also

compare.synds, compare.fit.synds

---

compare.fit.synds *Compare model estimates based on synthesised and observed data*

Description

The same model that was used for the synthesised data set is fitted to the observed data set. The coefficients with confidence intervals for the observed data is plotted together with their estimates from synthetic data. When more than one synthetic data set has been generated (object$m>1) combining rules are applied. Analysis-specific utility measures are used to evaluate differences between synthetic and observed data.

Usage

```
## S3 method for class 'fit.synds'
compare(object, data, plot = "Z",
  print.coef = FALSE, return.plot = TRUE, plot.intercept = FALSE,
  lwd = 1, lty = 1, lcol = c("#1A3C5A","#4187BF"),
  dodge.height = .5, point.size = 2.5, incomplete = FALSE,
  population.inference = FALSE, ci.level = 0.95, ...)

## S3 method for class 'compare.fit.synds'
print(x, print.coef = x$print.coef, ...)
```

Arguments

object	an object of type fit.synds created by fitting a model to synthesised data set using function glm.synds or lm.synds.
data	an original observed data set.
plot	values to be plotted: "Z" (Z scores) or "coef" (coefficients).
print.coef	a logical value determining whether tables of estimates for the original and synthetic data should be printed.
return.plot	a logical value indicating whether a confidence interval plot should be returned.
plot.intercept	a logical value indicating whether estimates for intercept should be plotted.
lwd	the line type.
lty	the line width.
lcol	line colours.
dodge.height	size of vertical shifts for confidence intervals to prevent overlaping.
point.size	size of plotting symbols used to plot point estimates of coefficients.
incomplete	a logical value indicating whether the method of Reiter (2003) for what he terms partially synthetic data should be used for inference. It requires multiple syntheses with m greater than the number of coefficients, ideally at least 5 more.

population.inference
: a logical value indicating whether intervals for inference to population quantities, as decribed by Karr et al. (2006), should be calculated and plotted. This option suppresses the lack-of-fit test and the standardised differences since these are based on differences standardised by the original interval widths.

ci.level
: Confidence interval coverage as a proportion.

...
: additional parameters passed to ggplot.

x
: an object of class compare.fit.synds.

Details

This function can be used to evaluate whether the method used for synthesis is appropriate for the fitted model. If this is the case the estimates from the synthetic dataof what would be expected from the original data xpct(Beta) xpct(Z) should not differ from the estimates from the observed data (Beta and Z) by more than would be expected from the standard errors (se(Beta) and se(Z)). For more details see the vignette on inference.

Value

An object of class compare.fit.synds which is a list with the following components:

call
: the original call to fit the model to the synthesised data set.

coef.obs
: a data frame including estimates based on the observed data: coefficients (Beta), their standard errors (se(Beta)) and Z scores (Z).

coef.syn
: a data frame including (combined) estimates based on the synthesised data: point estimates of observed data coefficients (B.syn), standard errors of those estimates (se(B.syn)), estimates of the observed standard errors (se(Beta).syn), Z scores estimates (Z.syn) and their standard errors (se(Z.syn)). Note that se(B.syn) and se(Z.syn) give the standard errors of the mean of the m syntheses and can be made very small by increasing m (see the vignette on inference for more details).

coef.diff
: a data frame containing standardized differences between the coefficients estimated from the original data and those calculated from the combined synthetic data. The difference is standardized by dividing by the estimated standard error of the fit from the original. The corresponding p-values are calculated from a standard Normal distribution and represent the probability of achieving differences as large as those found if the model use for synthesis is compatible with the model that generated the original data.

mean.abs.std.diff
: Mean absolute standardized difference (over all coefficients).

ci.overlap
: a data frame containing the percentage of overlap between the estimated synthetic confidence intervals and the original sample confidence intervals for each parameter. When population.inference = TRUE overlaps are calculated as suggested by Karr et al. (2006). Otherwise a simpler overlap measure with intervals of equal length is calculated.

mean.ci.overlap
: Mean confidence interval overlap (over all coefficients).

`lack.of.fit` lack-of-fit measure from all `m` synthetic data sets combined, calculated as follows, when `incomplete.method = FALSE`. The vector of mean differences (`diff`) between the coefficients calculated from the synthetic and original data provides a standardised `lack-of-fit = t(diff) %*% V^(-1) t(diff)`, where `%*%` represents the matrix product and `V^(-1)` is the inverse of the variance-covariance matrix for the mean coefficients from the synthetic data. If the model used to synthesize the data is correct this quantity, which is a Mahalanobis distance measure, will follow a chi-squared distribution with degrees of freedom, and thus expectation, equal to the number of parameters (`p`) in the fitted model. When `incomplete.method = TRUE` the function above follows a Hotelling's `T*2` distribution and the lack-of-fit statistic is referred to an `F(p, m - p)`.

`lof.pvalue` p-value for the combined lack-of-fit test of the NULL hypothesis that the method used for synthesis retains all relationships between variables that influence the parameters of the fit.

`ci.plot` `ggplot` of the the coefficients with confidence intervals for models based on observed and synthetic data. If `return.plot` was set to `FALSE` then `ci.plot` is `NULL`.

`print.coef` a logical value determining whether tables of estimates for the original and synthetic data should be printed.

`m` the number of synthetic versions of the original (observed) data.

`ncoef` the number of coefficients in the fitted model (including an intercept).

`incomplete` whether methods for incomplete synthesis due to Reiter (2003) have been used in calculations.

`population.inference` whether intervals as decribed by Karr et al. (2016) have been calculated.

References

Karr, A., Kohnen, C.N., Oganian, A., Reiter, J.P. and Sanil, A.P. (2006). A framework for evaluating the utility of data altered to protect confidentiality. *The American Statistician*, **60**(3), 224-232.

Nowok, B., Raab, G.M and Dibben, C. (2016). synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software*, **74**(11), 1-26. doi: 10.18637/jss.v074.i11.

Reiter, J.P. (2003) Inference for partially synthetic, public use microdata sets. *Survey Methodology*, **29**, 181-188.

See Also

`summary.fit.synds`

Examples

```
ods <- SD2011[,c("sex","age","edu","smoke")]
s1 <- syn(ods, m = 3)
f1 <- glm.synds(smoke ~ sex + age + edu, data = s1, family = "binomial")
compare(f1, ods)
compare(f1, ods, print.coef = TRUE, plot = "coef")
```

compare.synds *Compare univariate distributions of synthesised and observed data*

Description

Compare synthesised data set with the original (observed) data set using percent frequency tables and histograms. When more than one synthetic data set has been generated (object$m>1), by default pooled synthetic data are used for comparison.

Usage

```
## S3 method for class 'synds'
compare(object, data, vars = NULL, msel = NULL,
  breaks = 20, nrow = 2, ncol = 2, rel.size.x = 1,
  cols = c("#1A3C5A","#4187BF"), stat = "percents", ...)

## S3 method for class 'compare.synds'
print(x, ...)
```

Arguments

object	an object of class synds, which stands for 'synthesised data set'. It is typically created by function syn() and it includes object$m synthesised data set(s).
data	an original (observed) data set.
vars	variables to be compared. If vars is NULL (the default) all synthesised variables are compared.
msel	index or indices of synthetic data copies for which a comparison is to be made. If NULL pooled synthetic data copies are compared with the original data.
breaks	the number of cells for the histogram.
nrow	the number of rows for the plotting area.
ncol	the number of columns for the plotting area.
rel.size.x	a number representing the relative size of x-axis labels.
cols	bar colors.
stat	determines whether tables and plots present percentages stat = "percents", the default, or counts stat = "counts". If m > 1 and msel = NULL average counts for synthetic data are presented.
...	additional parameters.
x	an object of class compare.synds.

Details

Missing data categories for numeric variables are plotted on the same plot as non-missing values. They are indicated by miss. suffix.

Value

An object of class `compare.synds` which is a list including a list of comparative frequency tables (`tables`) and a ggplot object (`plots`) with bar charts/histograms. If multiple plots are produced they and their corresponding frequency tables are stored as a list.

References

Nowok, B., Raab, G.M and Dibben, C. (2016). synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software*, **74**(11), 1-26. doi: 10.18637/jss.v074.i11.

Examples

```
ods <- SD2011[ , c("sex","age","edu","marital","ls","income")]
s1  <- syn(ods)
compare(s1, ods, vars = "ls")
compare(s1, ods, vars = "income", stat = "counts", breaks = 10)
```

`glm.synds, lm.synds` *Fitting (generalized) linear models to synthetic data*

Description

Fits generalized linear models or simple linear models to the synthesised data set(s) using `glm` and `lm` function respectively.

Usage

```
glm.synds(formula, family = "binomial", data,  ...)
lm.synds(formula, data, ...)

## S3 method for class 'fit.synds'
print(x, msel = NULL, ...)
```

Arguments

`formula`	a symbolic description of the model to be estimated. A typical model has the form `response ~ predictors`. See the documentation of `glm` and `formula` for details.
`family`	a description of the error distribution and link function to be used in the model. See the documentation of `glm` and `family` for details.
`data`	an object of class `synds`, which stands for 'synthesised data set'. It is typically created by function `syn` and it includes `data$m` synthesised data set(s).
`...`	additional parameters passed to `glm` or `lm`.
`x`	an object of class `fit.synds`.
`msel`	index or indices of synthetic data copies for which coefficient estimates are to be displayed. If `NULL` (default) the combined (average) coefficient estimates are printed.

Value

An object of class `fit.synds`. It is a list with the following components:

`call`	the original call to `glm.synds` or `lm.synds`.
`mcoefavg`	combined (average) coefficient estimates.
`mvaravg`	combined (average) variance estimates of `mcoef`.
`analyses`	`summary.glm` or `summary.lm` object respectively or a list of `m` such objects.
`fitting.function`	function used to fit the model.
`n`	a number of cases in the original data.
`k`	a number of cases in the synthesised data.
`proper`	a logical value indicating whether synthetic data were generated using proper synthesis.
`m`	the number of synthetic versions of the observed data.
`method`	a vector of synthesising methods applied to each variable in the saved synthesised data.
`mcoef`	a matrix of coefficients estimates from all `m` syntheses.
`mvar`	a matrix of variance estimates from all `m` syntheses.

See Also

`glm`,`lm`, `multinom.synds`, `compare`

Examples

```
### Logit model
ods <- SD2011[1:1000, c("sex", "age", "edu", "marital", "ls", "smoke")]
s1 <- syn(ods, m = 3)
f1 <- glm.synds(smoke ~ sex + age + edu + marital + ls, data = s1, family = "binomial")
f1
print(f1, msel = 1:2)

### Linear model
ods <- SD2011[1:1000,c("sex", "age", "income", "marital", "depress")]
ods$income[ods$income == -8] <- NA
s2 <- syn(ods, m = 3)
f2 <- lm.synds(depress ~ sex + age + log(income) + marital, data = s2)
f2
print(f2,1:3)
```

multi.compare *Multivariate comparison of synthesised and observed data*

Description

Graphical comparisons of a variable (var) in the synthesised data set with the original (observed) data set within subgroups defined by the variables in a vector by. var can be a factor or a continuous variable and the plots produced will depend on the class of var. The variables in by will usually be factors or variables with only a few values.

Usage

```
multi.compare(object, data, var = NULL, by = NULL, msel = NULL,
  barplot.position = "fill", cont.type = "hist", y.hist = "count",
  boxplot.point = TRUE, binwidth = NULL, ...)
```

Arguments

object	an object of class synds, which stands for 'synthesised data set'. It is typically created by function syn() and it includes object$m synthesised data set(s).
data	an original (observed) data set.
var	variable to be compared between observed and synthetic data within subgroups.
by	variables to be tabulated or cross-tabulated to form groups.
barplot.position	type of barplot. The default "fill" gives a single bar with the proportions in each group while "dodge" gives side-by-side bars with the numbers in each category.
cont.type	default "hist" gives histograms and "boxplot" gives boxplots.
y.hist	defines y scale for histograms - "count" is default; "density" gives proportions.
boxplot.point	default (TRUE) adds individual points to boxplots.
msel	numbers of synthetic data sets to be used - must be numbers in the range 1:object$m - defaults to 1:object$m
binwidth	sets width of a bin for histograms.
...	additional parameters that can be supplied to ggplot.

Value

Plots as specified above. A table of the numbers in the subgroups is printed to the R console.

See Also

compare.synds, compare.fit.synds

Examples

```
### default synthesis of selected variables
vars <- c("sex", "age", "edu", "smoke")
ods  <- na.omit(SD2011[1:1000, vars])
s1 <- syn(ods)

### categorical var
multi.compare(s1, ods, var = "smoke", by = c("sex","edu"))

### numeric var
multi.compare(s1, ods, var = "age", by = c("sex"), y.hist = "density", binwidth = 5)
multi.compare(s1, ods, var = "age", by = c("sex", "edu"), cont.type = "boxplot")
```

`multinom.synds` *Fitting multinomial models to synthetic data*

Description

Fits multinomial models to the synthesised data set(s) using the `multinom` function.

Usage

```
multinom.synds(formula, data,  ...)
```

Arguments

`formula`	a symbolic description of the model to be estimated. A typical model has the form `response ~ predictors`. See the documentation of `multinom` and `formula` for details.
`data`	an object of class `synds`, which stands for 'synthesised data set'. It is typically created by function `syn` and it includes `data$m` synthesised data set(s).
`...`	additional parameters passed to `multinom`.

Value

An object of class `fit.synds`. It is a list with the following components:

`call`	the original call to `multinom.synds`.
`mcoefavg`	combined (average) coefficient estimates.
`mvaravg`	combined (average) variance estimates of `mcoef`.
`analyses`	an object summarising the fit to each synthetic data set or a list of `m` such objects. Note that this is different from the object created by `summary.multinom` to make it compatible with other fitting methods. In particular the coefficients are vectors, not matrices.
`fitting.function`	function used to fit the model.

n	a number of cases in the original data.
k	a number of cases in the synthesised data.
proper	a logical value indicating whether synthetic data were generated using proper synthesis.
m	the number of synthetic versions of the observed data.
method	a vector of synthesising methods applied to each variable in the saved synthesised data.
mcoef	a matrix of coefficients estimates from all m syntheses.
mvar	a matrix of variance estimates from all m syntheses.

See Also

multinom, glm.synds, compare

Examples

```
ods <- SD2011[1:1000, c("sex", "age", "edu", "marital", "ls", "smoke")]
s1 <- syn(ods, m = 3)
f1 <- multinom.synds(edu  ~ sex + age, data = s1)
summary(f1)
print(summary(f1, msel = 1:2))
compare(f1,ods)
```

---

numtocat.syn *Group numeric variables before synthesis*

Description

Make a new data frame with selected numeric variables grouped into factors with ranges selected from the data.

Usage

```
numtocat.syn(data, numtocat = NULL, print.flag = TRUE, cont.na = NULL,
             catgroups = 5, style.groups = "fisher")
```

Arguments

data	a data frame.
numtocat	a vector of numbers or variable names of numeric variables to be grouped into factors. If NULL all the numeric variables in data will be grouped.
print.flag	if TRUE a list of grouped variables is printed
cont.na	a named list that gives the values of the named variables to be treated as separate categories, often missing values like -8. See the corresponding parameter of syn().

catgroups	a single integer or a vector of integers indicating the target number of groups for the variables in numtocat in the same order as numtocat, or as their relative postions in data. The achieved number of groups may be different if, for example there are fewer than ngroups distinct values.
style.groups	parameter of the function classInt() that determines how the breaks used to categorise each variable are chosen. See the help file for classInt() for details

Value

a list with the following components:

data	a data frame with the numeric variables replaced by factors grouped into ranges.
breaks	a named list of the breaks used to divide each numeric variable into categories.
levels	a named list of the levels for the categories of each numeric variable.
orig	a data frame with the original numeric data.
cont.na	a named list of the levels for the categorical version of each numeric variable.
numtocat	names of the variables changed to categories.
ind	positions in data of the variables changed to categories.

Examples

```
SD2011.cat <- numtocat.syn(SD2011, cont.na = list(income = -8 , unempdur = -8,
nofriend = -8))
summary(SD2011.cat$data)
```

---

read.obs *Importing original data sets form external files*

Description

Imports data data sets form external files into a data frame. Currently supported files include: sav (SPSS), dta (Stata), xpt (SAS), csv (comma-separated file), tab (tab-delimited file) and txt (delimited text files). For SPSS, Stata and SAS it uses functions from the foreign package with some adjustments where necessary.

Usage

```
read.obs(file, convert.factors = TRUE, lab.factors = FALSE,
export.lab = FALSE, ...)
```

Arguments

file	the name of the file (including extension) which the data are to be read from.
convert.factors	a logical value indicating whether variables with value labels in Stata and SPSS should be converted into R factors with those levels.
lab.factors	a logical value indicating whether variables with complete value labels but imported using their numeric codes (convert.factors = FALSE) should be converted from numeric to factor variables.
export.lab	a logical variable indicating whether labels from SPSS or Stata should be exported to an external file.
...	additional parameters passed to read functions.

Value

A data frame with an imported data set. For SPSS, Stata and SAS it has attributes with labels.

See Also

write.syn

---

replicated.uniques *Replications in synthetic data*

Description

Determines which unique units in the synthesised data set(s) replicates unique units in the original observed data set.

Usage

```
replicated.uniques(object, data, exclude = NULL)
```

Arguments

object	an object of class synds, which stands for 'synthesised data set'. It is typically created by function syn() and it includes object$m synthesised data set(s).
data	the original observed data set.
exclude	a single string or a vector of strings with name(s) of variable(s) to be excluded from the identification of uniques.

Value

A list with the following components:

`replications` a vector (for `object$m = 1`) or a data frame with `object$m` columns (for `object$m > 1`) with logical values indicateing duplicates in `m`th synthetic data set.

`no.replications` a single number or a vector of `object$m` integers indicating the number of duplicates in the synthetic data set(s).

`no.uniques` a number of unique individuals in the original data set.

`per.replications` a single number or a vector of `object$m` numeric values indicating the percentage of duplicates in the synthetic data set(s).

See Also

sdc

Examples

```
ods <- SD2011[1:1000,c("sex","age","edu","marital","smoke")]
s1 <- syn(ods, m = 2)
replicated.uniques(s1,ods)
```

SD2011 *Social Diagnosis 2011 - Objective and Subjective Quality of Life in Poland*

Description

Sample of 5,000 individuals from the Social Diagnosis 2011 survey; selected variables only.

Usage

```
SD2011
```

Format

A data frame with 5,000 observations on the following 35 variables:

sex Sex

age Age of person, 2011

agegr Age group, 2011

placesize Category of the place of residence

region Region (voivodeship)

edu Highest educational qualification, 2011

eduspec Discipline of completed qualification

socprof Socio-economic status, 2011

unempdur Total duration of unemployment in the last 2 years (in months)

income Personal monthly net income

marital Marital status

mmarr Month of marriage

ymarr Year of marriage

msepdiv Month of separation/divorce

ysepdiv Year of separation/divorce

ls Perception of life as a whole

depress Depression symptoms indicator

trust View on interpersonal trust

trustfam Trust in own family members

trustneigh Trust in neighbours

sport Active engagement in some form of sport or exercise

nofriend Number of friends

smoke Smoking cigarettes

nociga Number of cigarettes smoked per day

alcabuse Drinking too much alcohol

alcsol Starting to use alcohol to cope with troubles

workab Working abroad in 2007-2011

wkabdur Total time spent on working abroad

wkabint Plans to go abroad to work in the next two years

wkabintdur Intended duration of working abroad

emcc Intended destination country

englang Knowledge of English language

height Height of person

weight Weight of person

bmi Body mass index

Note

Please note that the original variable names have been changed to make them more self-explanatory. Some variable labels have been adjusted as well.

Source

Council for Social Monitoring. Social Diagnosis 2000-2011: integrated database. http://www.diagnoza.com/index-en.html [downloaded on 13/12/2013]

References

Czapinski J. and Panek T. (Eds.) (2011). Social Diagnosis 2011. Objective and Subjective Quality of Life in Poland - full report. Contemporary Economics, Volume 5, Issue 3 (special issue) http://ce.vizja.pl/en/issues/volume/5/issue/3#art254

Examples

```
spineplot(englang ~ agegr, data = SD2011, xlab = "Age group", ylab = "Knowledge of English")
boxplot(income ~ sex, data = SD2011[SD2011$income != -8,])
```

---

sdc *Tools for statistical disclosure control (sdc)*

Description

Labeling and removing unique replicates of unique actual (observed) individuals.

Usage

```
sdc(object, data, label = NULL, rm.replicated.uniques = FALSE,
 uniques.exclude = NULL, recode.vars = NULL, bottom.top.coding = NULL,
 recode.exclude = NULL, smooth.vars = NULL)
```

Arguments

`object`	an object of class `synds`, which stands for 'synthesised data set'. It is typically created by function `syn()` and it includes `object$m` synthesised data set(s).
`data`	the original (observed) data set.
`label`	a single string with a label to be added to the synthetic data sets as a new variable to make it clear that the data are synthetic/fake.
`rm.replicated.uniques`	a logical value indicating whether unique replicates of units that are unique also in the orginal data set should be removed.
`uniques.exclude`	a single string or a vector of strings with name(s) of variable(s) to be excluded from the identification of uniques.
`recode.vars`	a single string or a vector of strings with name(s) of variable(s) to be bottom- or/and top-coded.
`bottom.top.coding`	a list of two-element vectors specifing bottom and top codes for each variable in `recode.vars`. If there is no need for bottom or top coding `NA` should be used. If only one variable is to be recoded, codes can be given as a two-element vector.
`recode.exclude`	a list specifying for each variable in `recode.vars` values to be excluded from recoding, e.g. missing data codes. If all values should be considered for recoding `NA` should be used. If only one variable is to be recoded, code(s) can be given as a single number or a vector.

smooth.vars — a single string or a vector of strings with name(s) of numeric variable(s) to be smoothed (smooth.spline function is used).

Value

An object provided as an argument adjusted in accordance with the other parameters' values.

See Also

replicated.uniques

Examples

```
ods <- SD2011[1:1000,c("sex","age","edu","marital","income")]
s1 <- syn(ods, m = 2)
s1.sdc <- sdc(s1, ods, label="false_data", rm.replicated.uniques = TRUE,
recode.vars = c("age","income"),
bottom.top.coding = list(c(20,80),c(NA,2000)),
recode.exclude = list(NA,c(NA,-8)))
```

---

summary.fit.synds *Inference from synthetic data*

Description

Combines the results of models fitted to each of the m synthetic data sets.

Usage

```
## S3 method for class 'fit.synds'
summary(object, population.inference = FALSE, msel = NULL,
  incomplete = FALSE, real.varcov = NULL, ...)

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

Arguments

object — an object of class fit.synds created by fitting a model to synthesised data set using function glm.synds or lm.synds.

population.inference — a logical value indicating whether inference should be made to population quantities. If FALSE inference is made to the results that would be expected from an analysis of the original data. This option should be selected if the synthetic data are being used for exploratory analysis, but the final published results will be obtained by running code on the original confidential data. If population.inference = TRUE results would allow population inference to be made from the synthetic data. In both cases the inference will depend on the

synthesising model being correct, but this can be checked by running the same analysis on the real data, see compare.fit.synds.

msel — index or indices of the synthetic datasets (1,...,m), for which summaries of fitted models are to be produced. If NULL (default) only the summary of combined estimates is produced.

incomplete — a logical value indicating whether inference is to use the method proposed by Reiter (2003) for what he terms partially synthetic data. This method is valid for any synthesis, but requires multiple syntheses. It is only necessary, when the dependent variable in a model is not completely synthesised and it only makes any difference to the results when population.inference = TRUE.

real.varcov — the estimated variance-covariance matrix of the fit of the model to the original data. This parameter is used in the function compare.fit.synds which has the original data as one of its parameters.

... — additional parameters.

x — an object of class summary.fit.synds.

Details

The mean of the estimates from each of the m synthetic data sets yields asymptotically unbiased estimates of the coefficients if the observed data conform to the distribution used for synthesis. The standard errors are estimated differently depending whether inference is made for the results that we would expect to obtain from the observed data or for the parameters of the population that we assume the observed data are sampled from. The standard errors also differ according to whether synthetic data were produced using simple or proper synthesis (for details see Raab et al. (2017)).

Value

An object of class summary.fit.synds which is a list with the following components:

call — the original call to glm.synds or lm.synds.

proper — a logical value indicating whether synthetic data were generated using proper synthesis.

population.inference — a logical value indicating whether inference is made to population coefficients or to the results that would be expected from an analysis of the original data (see above).

incomplete — a logical value indicating whether inference is to use the method proposed by Reiter (2003) for what he terms partially synthetic data.

fitting.function — function used to fit the model.

m — the number of synthetic versions of the original (observed) data.

coefficients — a matrix with combined estimates. If inference is required to the results that would be obtained from an analysis of the original data, (population.inference = FALSE) the coefficients are given by xpct(Beta), the standard errors by xpct(se.Beta) and the corresponding Z-statistic by xpct(Z). If the synthetic data are to be used to make inferences to population quantities (population.inference = TRUE),

	the coefficients are given by `Beta.syn`, their standard errors by `se.Beta.syn` and the Z-statistic by `Z.syn` (see vignette on inference for more details).
`n`	a number of cases in the original data.
`k`	the number of cases in the synthesised data. Note that if `k` and `n` are not equal and `population.inference = FALSE` (the default), then the standard errors produced will estimate what would be expected by an analysis of the original data set of size `n`.
`analyses`	`summary.glm` or `summary.lm` object respectively or a list of `m` such objects.
`msel`	index or indices of synthetic data copies for which summaries of fitted models are produced. If `NULL` only a summary of combined estimates is produced.

References

Nowok, B., Raab, G.M and Dibben, C. (2016). synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software*, **74**(11), 1-26. doi: 10.18637/jss.v074.i11.

Raab, G.M., Nowok, B. and Dibben, C. (2017). Practical data synthesis for large samples. *Journal of Privacy and Confidentiality*, **7**(3), 67-97. Available at: https://journalprivacyconfidentiality.org/index.php/jpc/article/view/407

Reiter, J.P. (2003) Inference for partially synthetic, public use microdata sets. *Survey Methodology*, **29**, 181-188.

See Also

`compare.fit.synds`, `summary`, `print`

Examples

```
ods <- SD2011[1:1000,c("sex","age","edu","ls","smoke")]

### simple synthesis
s1 <- syn(ods, m = 5)
f1 <- glm.synds(smoke ~ sex + age + edu + ls, data = s1, family = "binomial")
summary(f1)
summary(f1, population.inference = TRUE)

### proper synthesis
s2 <- syn(ods, m = 5, method = "parametric", proper = TRUE)
f2 <- glm.synds(smoke ~ sex + age + edu + ls, data = s2, family = "binomial")
summary(f2)
summary(f2, population.inference = TRUE)
```

summary.synds *Synthetic data object summaries*

Description

Produces summaries of the synthesised variables. When more than one synthetic data set has been generated (object$m>1), by default summaries are calculated by averaging summary values for all synthetic data copies (see msel argument).

Usage

```
## S3 method for class 'synds'
summary(object, msel = NULL, maxsum = 7,
digits = max(3, getOption("digits")-3), ...)

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

Arguments

object	an object of class synds; a result of a call to syn.
msel	index or indices of synthetic data copies for which a summary is desired. If NULL (default) summaries are calculated by averaging summary values for all synthetic data copies.
maxsum	integer, indicating how many levels should be shown for factors.
digits	integer, used for number formatting with format.
...	additional arguments passed to summary.
x	an object of class summary.synds.

Details

See summary for more details.

Value

An object of class summary.synds, which is a list with the following components:

m	the number of synthetic versions of the original (observed) data.
msel	index or indices of synthetic data copies for which a summary is produced. If NULL summaries are calculated by averaging summary values for all synthetic data copies.
method	a vector of synthesising methods applied to each variable in the saved synthesised data.
result	a table or a list of tabels (if more than one synthetic data set is selected) with summaries of synthesised variables.

References

Nowok, B., Raab, G.M and Dibben, C. (2016). synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software*, **74**(11), 1-26. doi: 10.18637/jss.v074.i11.

See Also

summary,print

Examples

```
s1 <- syn(SD2011[,c("sex","age","edu","marital")], m = 3)
summary(s1)
summary(s1, msel = c(1,3))
```

---

syn *Generating synthetic data sets*

Description

Generates synthetic version(s) of a data set. `syn.strata` performs stratified synthesis.

Usage

```
syn(data, method = vector("character", length = ncol(data)),
    visit.sequence = (1:ncol(data)), predictor.matrix = NULL,
    m = 1, k = nrow(data), proper = FALSE,
    minnumlevels = -1, maxfaclevels = 60,
    rules = NULL, rvalues = NULL,
    cont.na = NULL, semicont = NULL,
    smoothing = NULL, event = NULL, denom = NULL,
    drop.not.used = FALSE, drop.pred.only = FALSE,
    default.method = c("normrank", "logreg", "polyreg", "polr"),
    numtocat = NULL, catgroups = rep(5, length(numtocat)),
    models = FALSE, print.flag = TRUE, seed = "sample", ...)

syn.strata(data, strata = NULL,
    minstratumsize = 10 + 10 * length(visit.sequence),
    tab.strataobs = TRUE, tab.stratasyn = FALSE,
    method = vector("character", length = ncol(data)),
    visit.sequence = (1:ncol(data)), predictor.matrix = NULL,
    m = 1, k = nrow(data), proper = FALSE,
    minnumlevels = -1, maxfaclevels = 60,
    rules = NULL, rvalues = NULL,
    cont.na = NULL, semicont = NULL,
    smoothing = NULL, event = NULL, denom = NULL,
    drop.not.used = FALSE, drop.pred.only = FALSE,
    default.method = c("normrank", "logreg", "polyreg", "polr"),
```

```
    numtocat = NULL, catgroups = rep(5,length(numtocat)),
    models = FALSE, print.flag = TRUE, seed = "sample", ...)

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

Arguments

data	a data frame or a matrix (n x p) containing the original data. Observations are in rows and variables are in columns.
method	a single string or a vector of strings of length ncol(data) specifying the synthesising method to be used for each variable in the data. Order of variables is exactly the same as in data. If specified as a single string, the same method is used for all variables in a visit sequence unless a data type or a position in a visit sequence requires a different method. If method is set to "parametric" the default synthesising method specified by the default.method argument are applied. Variables that are transformations of other variables can be synthesised using a passive method that is specified as a string starting with ~ (see syn.passive). Variables that need not to be synthesised have the empty method "". By default all variables are synthesised using "cart" method, which is rpart implementation of a CART model (see syn.cart). See details for more information on method.
visit.sequence	a character vector of names of variables or an integer vector of their column indices specifying the order of synthesis. The default sequence 1:ncol(data) implies that column variables are synthesised from left to right. See details for more information.
predictor.matrix	a square matrix of size ncol(data) specifying the set of column predictors to be used for each target variable in the row. Each entry has value 0 or 1. A value of 1 means that the column variable is used as a predictor for the row variable. Order of variables is exactly the same as in data. By default all variables that are earlier in the visit sequence are used as predictors. For the default visit sequence (1:ncol(data)) the default predictor.matrix will have values of 1 in the lower triangle. See details for more information.
m	number of synthetic copies of the original (observed) data to be generated. The default is m = 1.
k	a size of the synthetic data set (k x p), which can be smaller or greater than the size of the original data set (n x p). The default is nrow(data) which means that the number of individuals in the synthesised data is the same as in the original (observed) data (k = n).
proper	a logical value with default set to FALSE. If TRUE proper synthesis is conducted.
minnumlevels	a minimum number of values a numeric variable should have to be treated as numeric. Numeric variables with fewer levels than minnumlevels are changed into factors. If set to -1 (default) numeric variables are left unchanged regardless of the number of values.
maxfaclevels	a maximum number of factor levels that can be handled. It can be increased but it may cause computational problems, especially for parametric methods.

`rules` a named list of rules for restricted values. Restricted values are those that are determined explicitly by values of other variables. The names of the list elements must correspond to the variables names for which the rules need to be specified.

`rvalues` a named list of the values corresponding to the rules specified by `rules`.

`cont.na` a named list of codes for missing values for continuous variables if different from the R missing data code `NA`. The names of the list elements must correspond to the variables names for which the missing data codes need to be specified.

`semicont` a named list of values at which semi-continuous variables have spikes. The names of the list elements must correspond to the names of the semi-continuous variables.

`smoothing` a named list specifying smoothing method (`"density"` or `""`) to be used for selected variables. Smoothing can only be applied to continuous variables synthesised using `sample`, `ctree`, `cart`, `normrank` or `nested` method. The names of the list elements must correspond to the names of the variables whose values are to be smoothed. Smoothing is applied to the synthesised values. For `"density"` smoothing a Gaussian kernel density estimator is applied with bandwidth selected using the Sheather-Jones 'solve-the-equation' method (see `bw.SJ`).

`event` a named list specifying for survival data the names of corresponding event indicators. The names of the list elements must correspond to the names of the survival variables.

`denom` a named list specifying for variables to be modelled using binomial regression the names of corresponding denominator variables. The names of the list elements must correspond to the names of the variables to be modelled using binomial regression.

`drop.not.used` a logical value. If `TRUE` (default) variables not used in synthesis are not saved in the synthesised data and are not included in the corresponding synthesis parameters.

`drop.pred.only` a logical value. If `TRUE` (default) variables not synthesised and used as predictors only are not saved in the synthesised data.

`default.method` a vector of four strings containing the default parametric synthesising methods for numerical variables, factors with two levels, unordered factors with more than two levels and ordered factors with more than two levels respectively. They are used when `method` is set to `"parametric"` or when there is an inconsistency between variable type and provided method.

`numtocat` a vector of numbers or names to indicate columns of `data` that are to be categorised into factors before synthesis. After the categorical variables have been synthesised the numerical variables are synthesised from them by the method `syn.nested` and are placed in the same position in the synthetic data as in the original. The categorised variables are not stored in the synthetic data. If you want to keep the categorised values you should change the relevant variables in `data` before running `syn` with the function `numtocat.syn()`

`catgroups` An integer or a vector of integers of the same length as `numtocat` giving the target number of groups into which of the numeric variables is to be categorised. The function `group_var` performs the categorisation.

`models` if `TRUE` parameters of models fitted to the original data and used to generate the synthetic values are stored.

print.flag	if TRUE (default) synthesising history and information messages will be printed at the console. For silent computation use print.flag = FALSE.
seed	an integer to be used as an argument for the set.seed(). If no integer is provided, the default "sample" will generate one and it will be stored. To prevent generating an integer set seed to NA.
...	additional arguments to be passed to synthesising functions. See section 'Details' below for more information.
strata	a numeric vector with strata identifiers or a string vector with names of stratifying variable(s).
minstratumsize	minimum size of each stratum.
tab.strataobs	a logical value indicating whether a frequency table of the number of observations in strata in the original data set should be printed.
tab.stratasyn	a logical value indicating whether a frequency table of the number of observations in strata in the synthetic data set(s) should be printed.
x	an object of class synds; a result of a call to syn.

Details

Only variables that are in visit.sequence with corresponding non-empty method are synthesised. The only exceptions are event indicators. They are synthesised along with the corresponding time to event variables and should not be included in visit.sequence. All other variables (not in visit.sequence or in visit.sequence with a corresponding blank method) can be used as predictors. Including them in visit.sequence generates a default predictor.matrix reflecting the order of variables in the visit.sequence otherwise predictor.matrix has to be adjusted accordingly. All predictors of the variables that are not in visit.sequence or are in visit.sequence but with a blank method are removed from predictor.matrix.

Variables to be synthesised that are not synthesised yet cannot be used as predictors. Also all variables used in passive synthesis or in restricted values rules (rules) have to be synthesised before the variables they apply to.

Mismatch between data type and synthesising method stops execution and print an error message but numeric variables with number of levels less than minnumlevels are changed into factors and methods are changed automatically, if necessary, to methods for categorical variables. Methods for variables not in a visit sequence will be changed into blank.

The built-in elementary synthesising methods defined by conditional distributions include:

ctree, cart classification and regression trees (CART), see syn.cart

bagging, random forests methods using ensembles of CART trees, see syn.bag and syn.rf

survctree classification and regression trees (CART) for duration time data (parametric methods for survival data are not implemented yet), see syn.survctree

norm normal linear regression, see syn.norm

normrank normal linear regression preserving the marginal distribution, see syn.normrank

lognorm, sqrtnorm, cubertnorm normal linear regression after natural logarithmic, square root and cube root transformation of a dependent variable respectively, see syn.lognorm

logreg logistic regression, see syn.logreg

polyreg unordered polytomous regression, see syn.polyreg

polr ordered polytomous regression, see syn.polr

pmm predictive mean matching, see syn.pmm

sample random sample from the observed data, see syn.sample

passive function of other synthesised data, see syn.passive

nested bootstrap sample within each category of the original grouping variable, see syn.nested

satcat bootstrap sample within each category of the crosstabulation of all the predictor variables, see syn.satcat

These methods use a group of variables that are synthesised together. They must always be together at the start of the visit sequence:

catall fit a saturated log-linear model, see syn.catall

ipf fit a log-linear model, defined by its margins, by iterative proportional fitting see syn.ipf

The functions corresponding to these methods are called syn.method, where method is a string with the name of a synthesising method. For instance a function corresponding to ctree function is called syn.ctree. A new synthesising method can be introduced by writing a function named syn.newmethod and then specifying method parameter of syn function as "newmethod".

In order to use "nested" sampling, method parameter of syn function has to be specified as "nested.varname", where "varname" is the name of the grouped (less detailed) variable, the only one used in nested synthesis. A variable synthesised using "nested" method is excluded from synthesising other variables except when used for "nested" method.

Additional parameters can be passed to synthesising methods as part of the dots argument. They have to be named using period-separated method and parameter name (method.parameter). For instance, in order to set a minbucket (minimum number of observations in any terminal node of a CART model) for a ctree synthesising method, ctree.minbucket has to be specified. The parameters are method-specific and will be used for all variables to be synthesised using that method. See help for syn.method for further details about the allowed parameters for a specific method.

Value

An object of class synds, which stands for 'synthesised data set'. It is a list with the following components:

call	an original call to syn.
m	number of synthetic versions of the original (observed) data.
syn	a data frame (for m = 1) or a list of m data frames (for m > 1) with synthetic data set(s).
method	a vector of synthesising methods applied to each variable in the saved synthesised data.
visit.sequence	a vector of column indices of the visiting sequence. The indices refer to the columns in the saved synthesised data.
predictor.matrix	a matrix specifying the set of predictors used for each variable in the saved synthesised data.

smoothing	a vector specifying smoothing methods applied to each variable in the saved synthesised data.
event	a vector of integers specifying for survival data the column indices for corresponding event indicators. The indices refer to the columns in the saved synthesised data.
denom	a vector of integers specifying for variables modelled using binomial regression the column indices for corresponding denominator variables. The indices refer to the columns in the saved synthesised data.
proper	a logical value indicating whether proper synthesis was conducted.
n	a number of cases in the original data.
k	a number of cases in the synthesised data.
rules	a list of rules for restricted values applied to the synthetic data.
rvalues	a list of the values corresponding to the rules specified by `rules`.
cont.na	a list of codes for missing values for continuous variables.
semicont	a list of values for semi-continuous variables at which they have spikes.
drop.not.used	a logical value indicating whether variables not used in synthesis are saved in the synthesised data and corresponding synthesis parameters.
drop.pred.only	a logical value indicating whether variables not synthesised and used as predictors only are saved in the synthesised data.
models	if `models = TRUE` a named list of estimates of models fitted to the original data and used to generate the synthetic values is returned from the `$fit` component of each method (e.g. `syn.cart()`). The list is ordered by the variables position in the data, and any models used to predict missing values are appended to the list.
seed	an integer used as a `set.seed()` argument.
var.lab	a vector of variable labels for data imported from SPSS using `read.obs()`.
val.lab	a list value labels for factors for data imported from SPSS using `read.obs()`.
obs.vars	a vector of all variable names in the observed data set.

Note

See package vignette for additional information.

References

Nowok, B., Raab, G.M and Dibben, C. (2016). synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software*, **74**(11), 1-26. doi: 10.18637/jss.v074.i11.

See Also

`compare.synds`, `summary.synds`

Examples

```
### selection of variables
vars <- c("sex","age","marital","income","ls","smoke")
ods  <- SD2011[1:1000, vars]

### default synthesis
s1 <- syn(ods)
s1

### synthesis with default parametric methods
s2 <- syn(ods, method = "parametric", seed = 1)
s2$method

### multiple synthesis of selected variables with customised methods
s3 <- syn(ods, visit.sequence = c(2, 1, 4, 5), m = 2,
          method = c("logreg","sample","","normrank","ctree",""),
          ctree.minbucket = 10)
summary(s3)
summary(s3, msel = 1:2)

### adjustment to the default predictor matrix
s4.ini <- syn(data = ods, visit.sequence = c(1, 2, 5, 3),
              m = 0, drop.not.used = FALSE)
pM.cor <- s4.ini$predictor.matrix
pM.cor["marital","ls"] <- 0
s4 <- syn(data = ods, visit.sequence = c(1, 2, 5, 3),
          predictor.matrix = pM.cor)

### handling missing values in continuous variables
s5 <- syn(ods, cont.na = list(income = c(NA, -8)))

### rules for restricted values - marital status of males under 18 should be 'single'
s6 <- syn(ods, rules = list(marital = "age < 18 & sex == 'MALE'"),
          rvalues = list(marital = 'SINGLE'), method = "parametric", seed = 1)
with(s6$syn, table(marital[age < 18 & sex == 'MALE']))
### results for default parametric synthesis without the rule
with(s2$syn, table(marital[age < 18 & sex == 'MALE']))

### synthesis with ipf for all variables
s7 <- syn(ods[, 1:3], method = "ipf", numtocat = "age")

### stratified synthesis
s8 <- syn.strata(ods, strata = "sex")
```

---

syn.bag *Synthesis with bagging*

Description

Generates univariate synthetic data using bagging. It uses randomForest function from the **randomForest** package with number of sampled predictors equal to number of all predictors.

Usage

```
syn.bag(y, x, xp, smoothing, proper = FALSE, ntree = 10, ...)
```

Arguments

y	an original data vector of length n.
x	a matrix (n x p) of original covariates.
xp	a matrix (k x p) of synthesised covariates.
smoothing	smoothing method for continuous variables.
proper	...
ntree	number of trees to grow.
...	additional parameters passed to randomForest.

Details

...

Value

A vector of length k with synthetic values of y.

References

...

See Also

syn, syn.rf, syn.cart, randomForest

---

syn.catall *Synthesis of a group of categorical variables from a saturated model*

Description

A saturated model is fitted to a table produced by cross-tabulating all the variables.

Usage

```
syn.catall(x, k, proper = FALSE, priorn = 1, structzero = NULL,
           maxtable = 1e8, ...)
```

Arguments

x	a data frame (n x p) of the set of original variables.
k	a number of rows in each synthetic data set - defaults to n.
proper	if proper = TRUE x is replaced with a bootstrap sample before synthesis, thus effectively sampling from the posterior distribution of the model, given the data.
priorn	the sum of the parameters of the Dirichelet prior which can be thought of as a pseudo-count giving the number of observations that inform prior knowledge about the parameters.
structzero	a named list of lists that defines which cells in the table are structural zeros and will remain as zeros in the synthetic data, by leaving their prior as zeros. Each element of the structzero list is a list that describes a set of cells in the table defined by a combination of two or more variables and a name of each such element must consist of those variable names seperated by an underscore, e.g. sex_edu. The length of each such element is determined by the number of variables and each component gives the variable levels (numeric or labels) that define the structural zero cells (see an example below).
maxtable	a number of cells in the cross-tabulation of all the variables that will trigger a severe warning.
...	additional parameters.

Details

When used in syn function the group of categorical variables with method = "catall" must all be together at the start of the visit.sequence. Subsequent variables in visit.sequence are then synthesised conditional on the synthesised values of the grouped variables. A saturated model is fitted to a table produced by cross-tabulating all the variables. Prior probabilities for the proportions in each cell of the table are specified from the parameters of a Dirichlet distribution with the same parameter for every cell in the table that is not a structural zero (see above). The sum of these parameters is priorn so that each one is $priorn/N$ where N is the number of cells in the table that are not structural zeros. The default priorn = 1 can be thought of as equivalent to the knowledge that 1 observation would be equally likely to be in any cell that is not a structural zero. The posterior expectation, given the observed counts, for the probability of being in a cell with observed count n_i is thus $(n_i + priorn/N)/(N + priorn)$. The synthetic data are generated from a multinomial distribution with parameters given by these probabilities.

Unlike syn.satcat, which fits saturated conditional models, the synthesised data can include any combination of variables, except those defined by the combinations of variables in structzero.

NOTE that when the function is called by setting elements of method in syn() to "catall", the parameters priorn, structzero and maxtable must be supplied to syn as e.g. catall.priorn.

Value

A list with two components. The first is a data frame (res) of dimension k x p containing the synthesised data and the second (fit) is the cross-tabulation of all the original variables used.

Examples

```
ods <- SD2011[, c(1, 4, 5, 6, 2, 10, 11)]
table(ods[, c("placesize", "region")])

# Each \code{placesize_region} sublist:
# for each relevant level of \code{placesize} defined in the first element,
# the second element defines regions (variable \code{region}) that do not
# have places of that size.

struct.zero <- list(
  placesize_region = list("URBAN 500,000 AND OVER", c(2, 4, 5, 8:13, 16)),
  placesize_region = list("URBAN 200,000-500,000", c(3, 4, 10:11, 13)),
  placesize_region = list("URBAN 20,000-100,000", c(1, 3, 5, 6, 8, 9, 14:15)))

syncatall <- syn(ods, method = c(rep("catall", 4), "ctree", "normrank", "ctree"),
                 catall.priorn = 2, catall.structzero = struct.zero)
```

---

syn.ctree, syn.cart *Synthesis with classification and regression trees (CART)*

Description

Generates univariate synthetic data using classification and regression trees (without or with bootstrap).

Usage

```
syn.ctree(y, x, xp, smoothing, proper = FALSE, minbucket = 5, mincriterion = 0.9, ...)
syn.cart(y, x, xp, smoothing, proper = FALSE, minbucket = 5, cp = 1e-08, ...)
```

Arguments

y	an original data vector of length n.
x	a matrix (n x p) of original covariates.
xp	a matrix (k x p) of synthesised covariates.
smoothing	smoothing method for continuous variables.
proper	for proper synthesis (proper = TRUE) a CART model is fitted to a bootstrapped sample of the original data.
minbucket	the minimum number of observations in any terminal node. See rpart.control and ctree_control for details.
cp	complexity parameter. Any split that does not decrease the overall lack of fit by a factor of cp is not attempted. Small values of cp will grow large trees. See rpart.control for details.
mincriterion	1 - p-value of the test that must be exceeded for a split to be retained. Small values of mincriterion will grow large trees. See ctree_control for details.
...	additional parameters passed to ctree_control for syn.ctree and rpart.control for syn.cart.

Details

The procedure for synthesis by a CART model is as follows:

1. Fit a classification or regression tree by binary recursive partitioning.
2. For each `xp` find the terminal node.
3. Randomly draw a donor from the members of the node and take the observed value of `y` from that draw as the synthetic value.

`syn.ctree` uses `ctree` function from the **party** package and `syn.cart` uses `rpart` function from the **rpart** package. They differ, among others, in a selection of a splitting variable and a stopping rule for the splitting process.

A Guassian kernel smoothing can be applied to continuous variables by setting smoothing parameter to `"density"`. It is recommended as a tool to decrease the disclosure risk. Increasing `minbucket` is another means of data protection.

CART models were suggested for generation of synthetic data by Reiter (2005) and then evaluated by Drechsler and Reiter (2011).

Value

A list with two components

`res`	A vector of length `k` with synthetic values of `y`.
`fit`	The fitted model which is an item of class `rpart.object` or `ctree.object` that can be printed or plotted.

References

Reiter, J.P. (2005). Using CART to generate partially synthetic, public use microdata. *Journal of Official Statistics*, **21**(3), 441–462.

Drechsler, J. and Reiter, J.P. (2011). An empirical evaluation of easily implemented, nonparametric methods for generating synthetic datasets. *Computational Statistics and Data Analysis*, **55**(12), 3232–3243.

See Also

`syn`, `syn.survctree`, `rpart`, `ctree`

`syn.ipf` *Synthesis of a group of categorical variables by iterative proportional fitting*

Description

A fit to the table is obtained from the log-linear fit that matches the numbers in the margins specified by the margin parameters.

Usage

```
syn.ipf(x, k, proper = FALSE, priorn = 1, structzero = NULL,
        gmargins = "twoway", othmargins = NULL,
        maxtable = 1e8, print.its = FALSE,  ...)
```

Arguments

x	a data frame of the set of original data to be synthesised.
k	a number of rows in each synthetic data set - defaults to n.
proper	if proper = TRUE x is replaced with a bootstrap sample before synthesis, thus effectively sampling from the posterior distribution of the model, given the data.
priorn	the sum of the parameters of the Dirichlet prior which can be thought of as a pseudo-count giving the number of observations that inform prior knowledge about the parameters.
structzero	a named list of lists that defines which cells in the table are structural zeros and will remain as zeros in the synthetic data, by leaving their prior as zeros. Each element of the structzero list is a list that describes a set of cells in the table defined by a combination of two or more variables and a name of each such element must consist of those variable names seperated by an underscore, e.g. sex_edu. The length of each such element is determined by the number of variables and each component gives the variable levels (numeric or labels) that define the structural zero cells (see an example below).
gmargins	a single character to define a group of margins. At present there is "oneway" and "twoway" option that creates, respectively, all 1-way and 2-way margins from the table.
othmargins	a list of margins that will be fitted. If gmargins is not NULL othmargins will be added to them.
maxtable	the number of cells in the cross-tabulation of all the variables that will trigger a severe warning.
print.its	if true the iterations from Ipfp will be printed on the console. Otherwise only a message as to whether the iterations have converged will be given at the end of the fitting.
...	additional parameters.

Details

When used in syn function the group of variables with method = "ipf" must all be together at the start of the visit sequence. This function is designed for categorical variables, but it can also be used for numerical variables if they are categorised by specifying them in the numtocat parameter of the main function syn. Subsequent variables in visit.sequence are then synthesised conditional on the synthesised values of the grouped variables. A fit to the table is obtained from the log-linear fit that matches the numbers in the margins specified by the margin parameters. Prior probabilities for the proportions in each cell of the table are given by a Dirichlet distribution with the same parameter for every cell in the table that is not a structural zero. The sum of these parameters is priorn. The default priorn = 1 can be thought of as equivalent to the knowledge that 1 observation would be equally likely to fall in any cell of the table. The synthetic data are generated from a multinomial

distribution with parameters given by the expected posterior probabilities for each cell of the table. If the maximum likelihood estimate from the log-linear fit to cell c_i is p_i and the table has N cells that are not structural zeros then the expectation of the posterior probability for this cell is $(p_i + priorn/N^2)/(1 + priorn/N^2)$ or equivalently $(N * p_i + priorn/N)/(N + priorn/N)$.

Unlike `syn.satcat`, which fits saturated models from their conditional distrinutions, `x` can include any combination of variables, including those not present in the original data, except those defined by `structzero`.

NOTE that when the function is called by setting elements of method in `syn` to `"ipf"`, the parameters `priorn`, `structzero`, `gmargins`, `othmargins`, `maxtable` and `print.its` must be supplied to syn as e.g. `ipf.priorn`.

Value

A list with two components. The first (`res`) is a data frame with `k` rows containing the synthesised data. The second (`fit`) is a list made up of two lists, the margins fitted and the original data for each margin.

Examples

```
ods <- SD2011[, c(1, 4, 5, 6, 2, 10, 11)]
table(ods[, c("placesize", "region")])

# Each \code{placesize_region} sublist:
# for each relevant level of \code{placesize} defined in the first element,
# the second element defines regions (variable \code{region}) that do not
# have places of that size.

struct.zero <- list(
  placesize_region = list("URBAN 500,000 AND OVER", c(2, 4, 5, 8:13, 16)),
  placesize_region = list("URBAN 200,000-500,000", c(3, 4, 10:11, 13)),
  placesize_region = list("URBAN 20,000-100,000", c(1, 3, 5, 6, 8, 9, 14:15)))

synipf <- syn(ods, method = c(rep("ipf", 4), "ctree", "normrank", "ctree"),
              ipf.gmargins = "twoway", ipf.othmargins = list(c(1, 2, 3)),
              ipf.priorn = 2, ipf.structzero = struct.zero)
```

---

syn.lognorm, syn.sqrtnorm, syn.cubertnorm

Synthesis by linear regression after transformation of a dependent variable

---

Description

Generates univariate synthetic data using linear regression of an outcome variable transformed by natural logarithm (`lognorm`), square root (`sqrtnorm`) or cube root (`cubertnorm`).

Usage

```
syn.lognorm(y, x, xp, proper = FALSE, ...)
syn.sqrtnorm(y, x, xp, proper = FALSE, ...)
syn.cubertnorm(y, x, xp, proper = FALSE, ...)
```

Arguments

y	an original data vector of length n.
x	a matrix (n x p) of original covariates.
xp	a matrix (k x p) of synthesised covariates.
proper	a logical value specifying whether proper synthesis should be conducted. See details.
...	additional parameters.

Details

Generates synthetic values using the spread around the fitted linear regression line of transformed y given x. For proper synthesis first the regression coefficients are drawn from normal distribution with mean and variance from the fitted model. The synthetic values are transformed back to the original scale.

Value

A vector of length k with synthetic values of y.

See Also

syn, syn.norm, syn.normrank

---

syn.logreg *Synthesis by logistic regression*

Description

Generates univariate synthetic data for binary or binomial response variable using logistic regression model.

Usage

```
syn.logreg(y, x, xp, denom = NULL, denomp = NULL, proper = FALSE, ...)
```

Arguments

y	an original data vector of length n.
x	a matrix (n x p) of original covariates.
xp	a matrix (k x p) of synthesised covariates.
denom	an original denominator vector of length n for a binomial regression model.
denomp	a synthesised denominator vector of length k for a binomial regression model.
proper	a logical value specifying whether proper synthesis should be conducted. See details.
...	additional parameters.

Details

Synthesis for binary response variables by the non-Bayesian or approximate Bayesian logistic regression model. The non-Bayesian method consists of the following steps:

1. Fit a logistic regression to the original data.
2. Calculate predicted inverse logits for synthesied covariates.
3. Compare the inverse logits to a random (0,1) deviate and get synthetic values.

The Bayesian version (for proper synthesis) includes additional step before computing inverse logits, namely drawing coefficients from normal distribution with mean and variance estimated in step 1.

The method relies on the standard `glm.fit` function. Warnings from `glm.fit` are suppressed. Perfect prediction is handled by the data augmentation method.

Value

A vector of length k with synthetic values (0 or 1) of y.

See Also

syn, glm, glm.fit

---

syn.nested *Synthesis for a variable nested within another variable.*

Description

Synthesizes one variable (y) from another one (x) when y is nested in the categories of x. A bootstrap sample is created from the original values of y within each category of xp (the synthesised values of the grouping variable).

Usage

```
syn.nested(y, x, xp, smoothing, cont.na,  ...)
```

Arguments

y	an original data vector of length n for the nested variable.
x	an original data vector of length n for the variable within which y is nested.
xp	a vector of length k with synthetic values of x.
smoothing	if this has the value "density" then values of a numeric y are smoothed after synthesis with a kernel density smoother.
cont.na	when y is numeric this can be a list or a vector giving values of y that indicate missing values.
...	additional parameters.

Details

An example would be when x is a classification of occupations and y is a more detailed sub-classification. It is intended that x is a categorical (factor) variable. A warning will be issued if the original y is not nested within x. A variable synthesised by syn.nested() is automatically excluded from predicting later variables because it will provide no extra information, given its grouping variable. syn.nested() is also used for the final synthesis of variables in syn() when the option numtocat is used to synthesise numerical variables as groups.

Value

A vector of length k with synthetic values of y.

---

syn.norm *Synthesis by linear regression*

Description

Generates univariate synthetic data using linear regression analysis.

Usage

```
syn.norm(y, x, xp, proper = FALSE, ...)
```

Arguments

y	an original data vector of length n.
x	a matrix (n x p) of original covariates.
xp	a matrix (k x p) of synthesised covariates.
proper	a logical value specifying whether proper synthesis should be conducted. See details.
...	additional parameters.

Details

Generates synthetic values using the spread around the fitted linear regression line of y given x. For proper synthesis first the regression coefficients are drawn from normal distribution with mean and variance from the fitted model.

Value

A vector of length k with synthetic values of y.

See Also

syn, syn.normrank, syn.lognorm

---

syn.normrank *Synthesis by normal linear regression preserving the marginal distribution*

Description

Generates univariate synthetic data using linear regression analysis and preserves the marginal distribution. Regression is carried out on Normal deviates of ranks in the original variable. Synthetic values are assigned from the original values based on the synthesised ranks that are transformed from their synthesised Normal deviates.

Usage

```
syn.normrank(y, x, xp, smoothing, proper = FALSE, ...)
```

Arguments

y	an original data vector of length n.
x	a matrix (n x p) of original covariates.
xp	a matrix (k x p) of synthesised covariates.
smoothing	smoothing method. See details.
proper	a logical value specifying whether proper synthesis should be conducted. See details.
...	additional parameters.

Details

First generates synthetic values of Normal deviates of ranks of the values in y using the spread around the fitted linear regression line of Normal deviates of ranks given x. Then synthetic Normal deviates of ranks are transformed back to get synthetic ranks which are used to assign values from y. For proper synthesis first the regression coefficients are drawn from normal distribution with mean and variance from the fitted model. A Guassian kernel smoothing can be applied by setting smoothing parameter to "density". It is recommended as a tool to decrease the disclosure risk.

Value

A vector of length k with synthetic values of y.

See Also

syn, syn.norm, syn.lognorm

---

syn.passive *Passive synthesis*

Description

Derives a new variable according to a specified function of synthesised data.

Usage

```
syn.passive(data, func)
```

Arguments

data	a data frame with synthesised data.
func	a formula specifying transformations on data. It is specified as a string starting with ~.

Details

Any function of the synthesised data can be specified. Note that several operators such as +, -, * and ^ have different meanings in formula syntax. Use the identity function I() if they should be interpreted as arithmetic operators, e.g. "~I(age^2)".

Value

A vector including the result of applying the formula.

Author(s)

Stef van Buuren, Karin Groothuis-Oudshoorn, 2000

References

Van Buuren, S. and Groothuis-Oudshoorn, K. (2011). mice: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software*, **45**(3), 1-67. http://www.jstatsoft.org/v45/i03/

See Also

syn

`syn.pmm` *Synthesis by predictive mean matching*

Description

Generates univariate synthetic data using predictive mean matching.

Usage

```
syn.pmm(y, x, xp, proper = FALSE, ...)
```

Arguments

`y`	an original data vector of length `n`.
`x`	a matrix (`n` x `p`) of original covariates.
`xp`	a matrix (`k` x `p`) of synthesised covariates.
`proper`	a logical value specifying whether proper synthesis should be conducted. See details.
`...`	additional parameters.

Details

Synthesis of `y` by predictive mean matching. The procedure is as follows:

1. Fit a linear regression to the original data.
2. Compute predicted values `y.hat` and `ysyn.hat` for the original `x` and synthesised `xp` covariates respectively.
3. For each predicted value `ysyn.hat` find donor observations with the closest predicted values `y.hat` (ties are broken by random selection), randomly sample one of them and take its observed value `y` as the synthetic value.

The Bayesian version (for proper synthesis) includes additional step before computing predicted values:

- Draw coefficients from normal distribution with mean and variance estimated in step 1 and use them to calculate predicted values for the synthesised covariates.

Value

A numeric vector of length `k` with synthetic values of `y`.

See Also

`syn`

`syn.polr` *Synthesis by ordered polytomous regression*

Description

Generates a synthetic categorical variable using ordered polytomous regression (without or with bootstrap).

Usage

```
syn.polr(y, x, xp, proper = FALSE, maxit = 1000, trace = FALSE,
  MaxNWts = 10000, ...)
```

Arguments

`y`	an original data vector of length `n`.
`x`	a matrix (`n` x `p`) of original covariates.
`xp`	a matrix (`k` x `p`) of synthesised covariates.
`proper`	for proper synthesis (`proper = TRUE`) a model is fitted to a bootstrapped sample of the original data.
`maxit`	the maximum number of iterations for `nnet`.
`trace`	switch for tracing optimization for `nnet`.
`MaxNWts`	the maximum allowable number of weights for `nnet`.
`...`	additional parameters passed to `optim` or `nnet`.

Details

Generates synthetic ordered categorical variables by the proportional odds logistic regression (polr) model. The function repeatedly applies logistic regression on the successive splits. The model is also known as the cumulative link model.

The algorithm of `syn.polr` uses the function `polr` from the **MASS** package.

In order to avoid bias due to perfect prediction, the data are augmented by the method of White, Daniel and Royston (2010).

In case the call to `polr` fails, usually because the data are very sparse, `multinom` function is used instead.

Value

A vector of length `k` with synthetic values of `y`.

References

White, I.R., Daniel, R. and Royston, P. (2010). Avoiding bias due to perfect prediction in multiple imputation of incomplete categorical variables. *Computational Statistics and Data Analysis*, **54**, 2267–2275.

See Also

syn,syn.polyreg multinom, polr

`syn.polyreg` *Synthesis by unordered polytomous regression*

Description

Generates a synthetic categorical variable using unordered polytomous regression (without or with bootstrap).

Usage

```
syn.polyreg(y, x, xp, proper = FALSE, maxit = 1000, trace = FALSE,
            MaxNWts = 10000, ...)
```

Arguments

y	an original data vector of length n.
x	a matrix (n x p) of original covariates.
xp	a matrix (k x p) of synthesised covariates.
proper	for proper synthesis (proper = TRUE) a multinomial model is fitted to a bootstrapped sample of the original data.
maxit	the maximum number of iterations for nnet.
trace	switch for tracing optimization for nnet.
MaxNWts	the maximum allowable number of weights for nnet.
...	additional parameters passed to nnet.

Details

Generates synthetic categorical variables by the polytomous regression model. The method consists of the following steps:

1. Fit categorical response as a multinomial model.
2. Compute predicted categories.
3. Add appropriate noise to predictions.

The algorithm of syn.polyreg uses the function multinom from the **nnet** package. Any numerical variables are scaled to cover the range (0,1) before fitting. Warnings are printed if the algorithm fails to converge in maxit iterations and also if the synthesised data has only one category. The latter may occur if the variable being synthesised is sparse so that the algorithm fails to iterate.

In order to avoid bias due to perfect prediction, the data are augmented by the method of White, Daniel and Royston (2010).

NOTE that when the function is called by setting elements of method in syn() to "polyreg", the parameters maxit, trace and MaxNWts can be supplied to syn() as e.g. polyreg.maxit.

Value

A vector of length k with synthetic values of y.

References

White, I.R., Daniel, R. and Royston, P. (2010). Avoiding bias due to perfect prediction in multiple imputation of incomplete categorical variables. *Computational Statistics and Data Analysis*, **54**, 2267–2275.

See Also

syn, syn.polr, multinom, polr

---

syn.rf *Synthesis with random forest*

Description

Generates univariate synthetic data using Breiman's random forest algorithm classification and regression. It uses randomForest function from the **randomForest** package.

Usage

```
syn.rf(y, x, xp, smoothing, proper = FALSE, ntree = 10, ...)
```

Arguments

y	an original data vector of length n.
x	a matrix (n x p) of original covariates.
xp	a matrix (k x p) of synthesised covariates.
smoothing	smoothing method for continuous variables.
proper	...
ntree	number of trees to grow.
...	additional parameters passed to randomForest.

Details

...

Value

A vector of length k with synthetic values of y.

References

...

See Also

syn, syn.bag, syn.cart, randomForest

`syn.sample` *Synthesis by simple random sampling*

Description

Generates a random sample from the observed data.

Usage

```
syn.sample(y, xp, smoothing, cont.na, proper = FALSE, ...)
```

Arguments

y	an original data vector of length n.
xp	a target length k of a synthetic data vector.
smoothing	smoothing method for a continous variable.
cont.na	a vector of codes for missing values for continuous variables that should be excluded from smoothing.
proper	if proper = TRUE values are sampled from a bootstrapped sample of the original data.
...	additional parameters passed to sample.

Details

A simple random sample with replacement is taken from the observed values in y and used as synthetic values. A Guassian kernel smoothing can be applied to continuous variables by setting smoothing parameter to "density". It is recommended as a tool to decrease the disclosure risk.

Value

A vector of length k with synthetic values.

See Also

syn

`syn.satcat` *Synthesis from a saturated model based on all combinations of the predictor variables.*

Description

Synthesises one variable (y) from all possible combinations of its precitors (x). A bootstrap sample is created from the original values of y within each unique combinations of of xp (the syntheisised values of the grouping variable).

Usage

```
syn.satcat(y, x, xp, proper = FALSE, ...)
```

Arguments

y	an original data vector of length n for the satcat variable.
x	a matrix (n x p) with the original predictor variables for y.
xp	a matrix (k x p) with synthetic values of x.
proper	if proper = TRUE x and y are replaced with a bootstrap sample before synthesis, thus effectively sampling from the posterior distribution of the model, given the data.
...	additional parameters.

Details

It is intended that the variables in x are categorical (factor) variables. If y is also a categorical variable syn.satcat will give the same results as fitting a saturated polychotomous regression model but will usually be much faster. syn.satcat will fail with an error message if previous syntheses have generated a combination of variables in xp that was not present in x. Use of the syn.catall method for grouped variables can overcome this.

Value

A vector of length k with synthetic values of y.

Examples

```
ods <- SD2011[, c("region", "sex", "agegr", "placesize")]

s1 <- syn(ods, method = c("sample", "cart", "satcat", "cart"))

## Not run:
### mostly fails because too many small categories
s2 <- syn(ods, method = c("sample", "cart", "cart", "satcat"))
## End(Not run)
```

`syn.survctree` *Synthesis of survival time by classification and regression trees (CART)*

Description

Generates synthetic event indicator and time to event data using classification and regression trees (without or with bootstrap).

Usage

```
syn.survctree(y, yevent, x, xp, proper = FALSE, minbucket = 5, ...)
```

Arguments

`y`	a vector of length `n` with original time data.
`yevent`	a vector of length `n` with original event indicator data.
`x`	a matrix (`n` x `p`) of original covariates.
`xp`	a matrix (`k` x `p`) of synthesised covariates.
`proper`	for proper synthesis (`proper = TRUE`) a CART model is fitted to a bootstrapped sample of the original data.
`minbucket`	the minimum number of observations in any terminal node. See `ctree_control` for details.
`...`	additional parameters passed to `ctree`.

Details

The procedure for synthesis by a CART model is as follows:

1. Fit a tree-structured survival model by binary recursive partitioning (the terminal nodes include Kaplan-Meier estimates of the survival time).
2. For each `xp` find the terminal node.
3. Randomly draw a donor from the members of the node and take the observed value of `yevent` and `y` from that draw as the synthetic values.

NOTE that when the function is called by setting elements of method in `syn()` to `"survctree"`, the parameter `minbucket` can be supplied to `syn()` as e.g. `survctree.minbucket`.

Value

A list with the following components:

`syn.time`	a vector of length `k` with synthetic time values.
`syn.event`	a vector of length `k` with synthetic event indicator values.

See Also

`syn`, `syn.ctree`

utility.gen *Distributional comparison of synthesised and observed data*

Description

Distributional comparison of synthesised data set with the original (observed) data set using propensity scores.

Usage

```
utility.gen(object, data, method = "logit", maxorder = 1,
            tree.method = "rpart", resamp.method = NULL,
            nperms = 50, cp = 1e-3, minbucket = 5, mincriterion = 0,
            vars = NULL, aggregate = FALSE, maxit = 200, ngroups = NULL,
            print.every = 10, digits = 2, print.zscores = FALSE, zthresh = 1.6,
            print.ind.results = TRUE, print.variable.importance = FALSE, ...)

## S3 method for class 'utility.gen'
print(x, digits = x$digits,
  print.zscores = x$print.zscores, zthresh = x$zthresh,
  print.ind.results = x$print.ind.results,
  print.variable.importance = x$print.variable.importance, ...)
```

Arguments

object	an object of class synds, which stands for 'synthesised data set'. It is typically created by function syn() and it includes object$m synthesised data set(s) as object$syn. This a single data set when object$m = 1 or a list of length object$m when object$m > 1.
data	the original (observed) data set.
method	a single string specifying the method for modeling the propensity scores. Method can be selected from "logit" and "cart".
maxorder	maximum order of interactions to be considered in "logit" method. For model without interactions 0 should be provided.
tree.method	implementation of "cart" method, can be "rpart" or "ctree".
resamp.method	method used for resampling estimate of the null pMSE, can be "perm" or "pairs". For pMSEs calculated with method "cart" it defaults to "perm" if all the variables from "object" in "vars" have been synthesised or to "pairs" if some have not (i.e. have method = "").
nperms	number of permutations for the permutation test to obtain the null distribution of the utility measure when resamp.method = "perm".
cp	complexity parameter for classification with tree.method "rpart". Small values grow bigger trees.
minbucket	minimum number of observations allowed in a leaf for classification when method = "cart".

mincriterion	criterion between 0 and 1 to use to control tree.method = "ctree" when the tree will not be allowed to split further. A value of 0.95 would be equivalent to a 5% significance test. Here we set it to 0 to effectively disable this test and grow large trees.
vars	variables to be included in the utility comparison. If none are specified all the variables in the synthesised data will be included.
aggregate	logical flag as to whether the data should be aggregated by collapsing identical rows before computation. This can lead to much faster computation when all the variables are categorical. Only works for method = "logit".
maxit	maximum iterations to use when method = "logit". If the model does not converge in this number a warning will suggest increasing it.
ngroups	target number of groups for categorisation of each numeric variable: final number may differ if there are many repeated values. If NULL (default) variables are not categorised into groups.
print.every	controls the printing of progress of resampling when resamp.method is not NULL. When print.every = 0 no progress is reported, otherwise the resample number is printed every print.every.
...	additional parameters passed to glm, rpart, or ctree.
x	an object of class utility.gen.
digits	number of digits to print in the default output, excluding pMSE values.
print.zscores	logical value as to whether z-scores for coefficients of the logit model should be printed.
zthresh	threshold value to use to suppress the printing of z-scores under +/- this value for method = "logit". If set to NA all z-scores are printed.
print.ind.results	logical value as to whether utility score results from individual syntheses should be printed.
print.variable.importance	logical value as to whether the variable importance measure should be printed when tree.method = "rpart".

Details

This function follows the method for evaluating the utility of masked data as given in Snoke et al. (forthcoming) and originally proposed by Woo et al. (2009). The original and synthetic data are combined into one dataset and propensity scores, as detailed in Rosenbaum and Rubin (1983), are calculated to estimate the probability of membership in the synthetic data set. The utility measure is based on the mean squared difference between these probabilities and the probability expected if the data did not distinguish the synthetic data from the original. The expected probability is just the proportion of synthetic data in the combined data set, 0.5 when the original and synthetic data have the same number of records.

Propensity scores can be modeled by logistic regression method = "logit" or by two different implementations of classification and regression trees as method "cart". For logistic regression the predictors are all variables in the data and their interactions up to order maxorder. The default

of 1 gives all main effects and first order interactions. For logistic regression the null distribution of the propensity score is derived and is used to calculate ratios and standardised values.

For `method = "cart"` the expectation and variance of the null distribution is calculated from a permutation test.

If missing values exist, indicator variables are added and included in the model as recommended by Rosenbaum and Rubin (1984). For categorical variables, NA is treated as a new category.

Value

An object of class `utility.gen` which is a list including the utility measures their expected null values for each synthetic set with the following components:

`call`	the call that produced the result.
`m`	number of synthetic data sets in object.
`method`	method used to fit propensity score.
`tree.method`	cart function used to fit propensity score when `method = "cart"`.
`pMSE`	Propensity score mean square error from the utility model or a vector of these values if `object$m > 1`.
`utilVal`	utility value(s). Calculated from the `pMSE` as `pMSE*(n1+n2)^3/n1^2/n2` or a vector of these values if `object$m > 1`. For `method = "logit"` the null distribution of this quantity will be chi-squared with degrees of freedom equal to the number of parameters involving synthesised variables in the propensity score minus 1. For `method = "cart"` its distribution will be obtained by resampling.
`utilExp`	expected value(s) of the utility score if the synthesis method is correct.
`utilR`	ratio(s) of `utilVal(s)` to `utilExp`.
`utilStd`	utility value standardised by expressing it as z-scores, difference(s) from the expected value divided by the expected standard deviation.
`pval`	p-value(s) for the chi-square test(s) for `utilVal` with `utilExp` degrees of freedom.
`fit`	the fitted model for the propensity score or a list of fitted models of length `m` if `m > 0`.

References

Woo, M-J., Reiter, J.P., Oganian, A. and Karr, A.F. (2009). Global measures of data utility for microdata masked for disclosure limitation. *Journal of Privacy and Confidentiality*, **1**(1), 111-124.

Rosenbaum, P.R. and Rubin, D.B. (1984). Reducing bias in observational studies using subclassification on the propensity score. *Journal of the American Statistical Association*, **79**(387), 516-524.

Snoke, J., Raab, G.M., Nowok, B., Dibben, C. and Slavkovic, A. (2018). General and specific utility measures for synthetic data. *Journal of the Royal Statistical Society: Series A*, **181**, Part 3, 663-688.

See Also

`utility.tab`

Examples

```
## Not run:
  ods <- SD2011[1:1000, c("age", "bmi", "depress", "alcabuse", "englang")]
  s1 <- syn(ods, m = 5)
  utility.gen(s1, ods)
  u1 <- utility.gen(s1, ods)
  print(u1, print.zscores = TRUE, usethresh = TRUE)
  u2 <- utility.gen(s1, ods, groups = TRUE)
  print(u2, print.zscores = TRUE)
  u3 <- utility.gen(s1, ods, method = "cart", nperms = 20)
  print(u3, print.variable.importance = TRUE)
## End(Not run)
```

---

utility.tab *Tabular utility*

Description

Produce tables from observed and synthesized data and calculates utility measures to compare them with their expectation if the synthesising model is correct.

Usage

```
utility.tab(object, data, vars = NULL, ngroups = 5, useNA = TRUE,
            print.tables = length(vars) < 4, print.stats = 'VW',
            print.zdiff = FALSE, digits = 2, ...)

## S3 method for class 'utility.tab'
print(x, print.tables = x$print.tables,
  print.zdiff = x$print.zdiff, print.stats = x$print.stats,
  digits = x$digits, ...)
```

Arguments

object	an object of class synds, which stands for 'synthesised data set'. It is typically created by function syn() or syn.strata() and it includes object$m number of synthesised data set(s), as well as object$syn the synthesised data set, if m = 1, or a list of m such data sets.
data	the original (observed) data set.
vars	a single string or a vector of strings with the names of variables to be used to form the table.
ngroups	if numerical (non-factor) variables are included they will be classified into this number of groups to form tables. Classification is performed using classIntervals() function for n = ngroups. By default, to avoid problems for variables with a small number of unique values, style = "fisher". Arguments of classIntervals() may be, however, specified in the call to utility.tab().

useNA	determines if NA values are to be included in tables.
print.tables	a logical value that determines if tables of observed and synthesised are to be printed.
print.stats	Determines which chi-squred statistics to print to compare the observed and synthetic tables : 'VW' for Voas Williams, 'FT' for Freeman Tukey or c('VW','FT') for both.
print.zdiff	a logical value that determines if tables of Z scores for differences between observed and expected are to be printed.
digits	an integer indicating the number of decimal places for printing statistics, tab.zdiff and mean results for m > 1.
...	additional parameters; can be passed to classIntervals() function.
x	an object of class utility.tab.

Details

Forms tables of observed and synthesised values for the variables specified in vars. Two utility measures are calculated from the cells of the tables, a measure of fit proposed by Voas and Williams sum((observed-synthesied)^2/[(observed + synthesised)/2)]) and one proposed by Freeman and Tukey 4*sum((observed^(0.5)-synthesised^(0.5))^2)). In both cases those cells where observed and synthesised are both zero do not contribute to the sum. If the synthesising model is correct both of these measures should have chi-square distributions for large samples.

Value

An object of class utility.tab which is a list with the following components:

m	number of synthetic data sets in object, i.e. object$m.
tab.obs	a table from the observed data.
UtabFT	a vector with object$m values for the Freeman Tukey utility measure.
UtabVW	a vector with object$m values for the Voas Williamson utility measure.
df	a vector of degrees of freedom for the chi-square tests which equal to one minus the number of cells in the table with any observed or synthesised counts.
ratioFT	a vector with ratios of UtabFT to df.
ratioVW	a vector with ratios of UtabVW to df.
pvalFT	a vector with object$m p-values for the chi-square tests for the Freeman Tukey utility measure.
pvalVW	a vector with object$m p-values for the chi-square tests for the Voas Williamson utility measure.
nempty	a vector of length object$m with number of cells not contributing to the statistics.
tab.obs	a table from the observed data.
tab.syn	a table or a list of m tables from the synthetic data.
tab.zdiff	a table or a list of m tables of Z statistics for differences between observed and synthesised cells of the tables. Large absolute values indicate a large contribution to lack-of-fit.
n	number of observation in the original dataset.

References

Nowok, B., Raab, G.M and Dibben, C. (2016). synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software*, **74**(11), 1-26. doi: 10.18637/jss.v074.i11.

Read, T.R.C. and Cressie, N.A.C. (1988) *Goodness–of–Fit Statistics for Discrete Multivariate Data*, Springer–Verlag, New York.

Voas, D. and Williamson, P. (2001) Evaluating goodness-of-fit measures for synthetic microdata. *Geographical and Environmental Modelling*, **5**(2), 177-200.

See Also

utility.gen

Examples

```
ods <- SD2011[1:1000, c("sex", "age", "edu", "marital")]

s1 <- syn(ods, m = 10)
utility.tab(s1, ods, vars = c("marital", "sex"))

s2 <- syn(ods, m = 1)
utility.tab(s2, ods, vars = c("marital", "age"), ngroups = 3, print.tables = TRUE)
u2 <- utility.tab(s2, ods, vars = c("marital", "age"), style = "pretty")
print(u2, print.tables = TRUE, print.zdiff = TRUE)
```

---

write.syn *Exporting synthetic data sets to external files*

Description

Exports synthetic data set(s) from synthesised data set (synds) object to external files of selected format. Currently supported file formats include: SPSS, Stata, SAS, csv, tab, rda, RData and txt. For SPSS, Stata and SAS it uses functions from the foreign package with some adjustments where necessary. Information about the synthesis is written into a separate text file.

NOTE: Currently numeric codes and labels can be preserved correctly only for SPSS files imported into R using read.obs function.

Usage

```
write.syn(object, filename,
filetype = c("SPSS", "Stata", "SAS", "csv", "tab", "rda", "RData", "txt"),
convert.factors = "numeric", data.labels = NULL, save.complete = TRUE,
extended.info = TRUE, ...)
```

Arguments

object	an object of class synds, which stands for 'synthesised data set'. It is typically created by function syn and it includes object$m synthesised data set(s).
filename	the name of the file (excluding extension) which the synthetic data are to be written into. For multiple synthetic data sets it will be used as a prefix folowed respectively by _1, _..., _m.
filetype	a desired format of the output files.
convert.factors	a single string indicating how to handle factors in Stata output files. The default value is set to "numeric" in order to preserve the numeric codes from the original data. See write.dta for other possible values.
data.labels	a list with variable labels and value labels.
save.complete	a logical value indicating whether a complete 'synthesised data set' (synds) object should be saved into a file (synobject_filename.RData).
extended.info	a logical value indicating whether extended information should be saved into an information file.
...	additional parameters passed to write functions.

Value

File(s) with synthesised data set(s) and a text file with information about synthesis are produced. Optionally a complete synthesised data set object is saved into synobject_filename.RData file.

See Also

read.obs

Index