Package 'synthpop'

July 8, 2015

Type Package

Title Generating Synthetic Versions of Sensitive Microdata for Statistical Disclosure Control

Version 1.1-1

Date 2015-07-07

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.

License GPL-2 | GPL-3

Depends lattice, MASS, methods, nnet, ggplot2, coefplot

Imports graphics, stats, utils, rpart, party, foreign, plyr, proto

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2015-07-08 13:22:09

R topics documented:

synthpop-package 2
compare 3
compare.fit.synds 4
compare.synds 5

glm.synds, lm.synds . . . 6
read.obs . . . 8
replicated.uniques . . . 9
SD2011 . . . 10
sdc . . . 11
summary.fit.synds . . . 12
summary.synds . . . 14
syn . . . 16
syn.ctree, syn.cart . . . 21
syn.lognorm, syn.sqrtnorm, syn.cubertnorm . . . 22
syn.logreg . . . 23
syn.norm . . . 24
syn.normrank . . . 25
syn.passive . . . 26
syn.pmm . . . 27
syn.polr . . . 28
syn.polyreg . . . 29
syn.sample . . . 30
syn.survctree . . . 31
write.syn . . . 32

Index **33**

`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:	1.1-1
Date:	2015-07-07
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>

`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.

Usage

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

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

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).
`return.result`	a logical value indicating whether a table of estimates should be returned.
`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.
`...`	additional parameters passed to `ggplot`.
`x`	an object of class `compare.fit.synds`.

Details

This function can be used to evaluate whether the model used for synthesis is appropriate for the fitted model. If this is the case the estimates from the synthetic data (`B.syn` and `Z.syn`) should not differ from the estimates from the observed data (`Beta` and `Z`) by more than would be expected from the standard errors (`se(B.syn)` and `se(Z.syn)`).

Value

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

`fit.synds.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`.
`ci.plot`	`ggplot` of the the coefficients with confidence intervals for models based on observed and synthetic data.

If `return.result` was set to `FALSE`, `coef.obs` and `coef.obs` are both `NULL`. If `return.plot` was set to `FALSE`, `ci.plot` is `NULL`.

See Also

`summary.fit.synds`

Examples

```
ods <- SD2011[,c("sex","age","edu","smoke")]
s1 <- syn(ods, m = 5)
f1 <- glm.synds(smoke ~ sex + age + edu, data = s1, family = "binomial")
compare(f1, ods)
compare(f1, ods, 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"), ...)

## 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.
...	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 percent 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.

Examples

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

---

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.
proper	a logical value indicating whether synthetic data were generated using proper synthesis.
m	the number of synthetic versions of the observed data.
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.
mcoef	a matrix of coefficients estimates from all m syntheses.
mvar	a matrix of variance estimates from all m syntheses.

See Also

glm,lm

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)
```

`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)
```

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.

Value

A list with the following components:

replications	a vector (for object$m = 1) or a data frame with object$m columns (for object$m > 1) with logical values indicateing duplicates in mth synthetic data set.
no.replications	a single number or a vector of object$m integers indicating the number of duplicates in the synthetic data set(s).
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,
 recode.vars = NULL, bottom.top.coding = NULL, recode.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.
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.
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.

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, ...)

## 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 original data quantities.
msel	index or indices of synthetic data copies for which summaries of fitted models are to be produced. If NULL (default) a summary of combined estimates is produced.
...	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 would be obtained 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. (submitted 2014)).

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 to population coefficients or to coefficients of the actual (observed) data is made.
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. It includes point estimates of coefficients (B.syn), their standard errors (se(B.syn)) and Z scores (Z.syn) for population and observed data quantities respectively. For inference to original data quantities it contains in addition estimates of the actual standard errors based on synthetic data (se(Beta).syn) and standard errors of Z scores (se(Z.syn)).

n	a number of cases in the original data.
k	a number of cases in the synthesised data.
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 a summary of combined estimates is produced.

References

Raab, G.M., Nowok, B. and Dibben, C. (submitted 2014). A simplified approach to synthetic data. http://arxiv.org/abs/1409.0217

See Also

summary,print

Examples

```
ods <- SD2011[1:2000,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, 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.

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.

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 = 5,
    maxfaclevels = 60, rules = NULL, rvalues = NULL,
    cont.na = NULL, semicont = NULL, smoothing = NULL,
    event = NULL, denom = NULL, drop.not.used = TRUE, drop.pred.only = FALSE,
    default.method = c("normrank", "logreg", "polyreg", "polr"),
    diagnostics = 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 `~`. Variables that need not to be synthesised have the empty method `""`. By default all variables are synthesised using `ctree` implementation of a CART model. See details for more information.
`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.

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 or normrank 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 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.
diagnostics	a logical value. If TRUE diagnostic information are appended to the value of the function. If FALSE (default) only the synthesised data are saved.
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.
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 include:

ctree, cart classification and regression trees (CART)

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

norm normal linear regression

normrank normal linear regression preserving the marginal distribution

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

logreg logistic regression

polyreg unordered polytomous regression

polr ordered polytomous regression

pmm predictive mean matching

sample random sample from the observed data

passive function of other synthesised data

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"`.

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.
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().

Note

See package vignette for additional information.

See Also

compare.synds, summary.synds

Examples

```
### selection of variables
vars <- c("sex","age","marital","income","ls","smoke")
ods  <- SD2011[1:2000,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']))
```

`syn.ctree, syn.cart` *Synthesis by 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, ...)
syn.cart(y, x, xp, smoothing, proper = FALSE, minbucket = 5, cp = 1e-04, ...)
```

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. See `rpart.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 vector of length `k` with synthetic values of `y`.

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.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, ...)
```

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:

- Draw 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.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., 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 = 100, 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 = 100, 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.

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

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.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.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.

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`

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.
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.rda).
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.rda file.

See Also

read.obs

Index

∗Topic **datagen**
    syn, 16
    syn.ctree, syn.cart, 21
    syn.lognorm, syn.sqrtnorm, syn.cubertnorm, 22
    syn.logreg, 23
    syn.norm, 24
    syn.normrank, 25
    syn.passive, 26
    syn.pmm, 27
    syn.polr, 28
    syn.polyreg, 29
    syn.sample, 30
    syn.survctree, 31
∗Topic **datasets**
    SD2011, 10
∗Topic **manip**
    sdc, 11
∗Topic **multivariate**
    glm.synds, lm.synds, 6
∗Topic **package**
    synthpop-package, 2
∗Topic **regression**
    syn, 16
∗Topic **tree**
    syn, 16

bw.SJ, *17*

compare, 3
compare.fit.synds, *2, 3*, 4
compare.synds, *2, 3*, 5, *20*
ctree, *22*, *31*
ctree_control, *21*, *31*

family, *7*
format, *15*
formula, *7*

ggplot, *4*

glm, *6, 7*, *24*
glm.fit, *24*
glm.synds, *2*, *4*, *13*
glm.synds *(*glm.synds, lm.synds*)*, 6
glm.synds, lm.synds, 6

lm, *6, 7*
lm.synds, *2*, *4*, *13*
lm.synds *(*glm.synds, lm.synds*)*, 6

multinom, *28–30*

nnet, *28, 29*

optim, *28*

polr, *28–30*
print, *14, 15*
print.compare.fit.synds *(*compare.fit.synds*)*, 4
print.compare.synds *(*compare.synds*)*, 5
print.fit.synds *(*glm.synds, lm.synds*)*, 6
print.summary.fit.synds *(*summary.fit.synds*)*, 12
print.summary.synds *(*summary.synds*)*, 14
print.synds *(*syn*)*, 16

read.obs, 8, *32*
replicated.uniques, 9, *12*
rpart, *22*
rpart.control, *21*

SD2011, 10
sdc, *9*, 11
summary, *14, 15*
summary.fit.synds, *5*, 12
summary.synds, 14, *20*
syn, *2*, *7*, *15*, 16, *22–27*, *29–32*
syn.cart *(*syn.ctree, syn.cart*)*, 21
syn.ctree, *31*
syn.ctree *(*syn.ctree, syn.cart*)*, 21

syn.ctree, syn.cart, 21
syn.cubertnorm (syn.lognorm, syn.sqrtnorm, syn.cubertnorm), 22
syn.lognorm, *24*, *25*
syn.lognorm (syn.lognorm, syn.sqrtnorm, syn.cubertnorm), 22
syn.lognorm, syn.sqrtnorm, syn.cubertnorm, 22
syn.logreg, 23
syn.norm, *23*, 24, *25*
syn.normrank, *23*, *24*, 25
syn.passive, 26
syn.pmm, 27
syn.polr, 28, *30*
syn.polyreg, *29*, 29
syn.sample, 30
syn.sqrtnorm (syn.lognorm, syn.sqrtnorm, syn.cubertnorm), 22
syn.survctree, *22*, 31
synthpop (synthpop-package), 2
synthpop-package, 2

write.dta, *32*
write.syn, *9*, 32