Package 'synthpop'

August 18, 2014

Type Package

Title Generating synthetic versions of sensitive microdata for statistical disclosure control

Version 1.0-0

Date 2014-08-03

Author Beata Nowok, Gillian M Raab and Chris Dibben (first two authors in alphabetical order)

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

Imports rpart, party

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2014-08-18 17:15:24

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:	1.0-0
Date:	2014-08-03
License:	GPL-2 | GPL-3

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

Author(s)

Beata Nowok, Gillian M Raab and Chris Dibben (first two authors in alphabetical order) based on package **mice** (2.18) by Stef van Buuren and Karin Groothuis-Oudshoorn

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

---

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

Description

The same model that was used for the synthesised data set is fitted to the real data set. The coefficients with confidence intervals for both model estimates are plotted for comparison. When more than one synthetic data set has been generated (object$m>1) average of the means and standard errors are used.

Usage

```
compare.fit.synds(object, real.data, plot = "Z",
  return.result = TRUE, plot.intercept = FALSE, col = 1:2, ...)

## 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.
real.data	the original (real) data set.
plot	values to be plotted: "Z", "coefficients" or "both".
return.result	a logical value indicating whether a table of estimates should be printed.
plot.intercept	a logical value indicating whether estimates for intercept should be plotted.
col	vector of colours for lines and points for plotting real and synthesised estimates.
...	additional parameters passed to plot.
x	an object of class compare.fit.synds.

Value

Plot(s) of the coefficients with confidence intervals for both models.

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.
coefficients	a data frame including coefficients, Z-scores and standard errors for models fitted to real and synthetic data sets and synthesising errors for coefficients and Z-scores.

See Also

summary.fit.synds

Examples

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

---

compare.synds *Compare synthesised and real data*

Description

Compare synthesised data set with the original (real) data set using percent frequency tables and histograms. When more than one synthetic data set has been generated (object$m>1), only the first one is used for comparison.

Usage

```
compare.synds(object, data, vars = NULL, nrows = 1, ncols = 1, breaks = 50, ...)
```

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 (real) data set.
vars	variables to be compared. If vars is NULL (the default) all synthesised variables are compared.
nrows	the number of rows for the plotting area.
ncols	the number of columns for the plotting area.
breaks	the number of cells for the histogram.
...	additional parameters passed to barplot.

Value

Comparative percent frequency tables and histograms.

Examples

```
rds <- SD2011[ ,c("sex","age","edu","marital","ls","income")]
s1  <- syn(rds)
compare.synds(s1, rds, vars = "ls")
compare.synds(s1, rds, 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", object,  ...)
lm.synds(formula, object, ...)

## S3 method for class 'fit.synds'
print(x, msel = 1, ...)
```

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.
`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).
`...`	additional parameters passed to `glm` or `lm` .
`x`	an object of class `fit.synds`.
`msel`	index or indices of synthetic data copies for which results are to be displayed.

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`.
`proper`	a logical value indicating whether synthetic data were generated using proper synthesis.
`m`	the number of synthetic versions of the real 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.

See Also

glm,lm

Examples

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

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

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,])
```

`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, populationInference = FALSE, ...)

## 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`.
`populationInference`	a logical value indicating whether inference should be made to population quantities. If `FALSE` inference is made to original data quantities.
`...`	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 unbaised estimates of the coefficients. The variance is estimated differently depending whether inference is made to the real data quantities or the population parameters and whether synthetic data were produced using simple or proper synthesis (for details see Raab et al. (2014); expressions used to calculate variance for different cases are presented in Table 1).

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.
`fitting.function`	function used to fit the model.

m	the number of synthetic versions of the real data.
coefficients	a matrix with combined estimates. For inference to original data quantities it includes coefficients (beta syn) and their standard errors (se beta syn), Z scores (Z syn) and synthesising errors for coefficients (syn err beta) and Z-scores (syn err Z). For inference to population quantities it includes coefficients (est), synthesising errors (se) and Z scores (Z).
n	a number of cases in the original data.
k	a number of cases in the synthesised data.

References

Raab, G.M., Nowok, B., Dibben, C. (2014). A simplified approach to synthetic data. *Submitted*.

See Also

summary,print

Examples

```
rds <- SD2011[1:2000,c("sex","age","edu","ls","smoke")]

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

### proper synthesis
s2 <- syn(rds, m = 5, proper = TRUE)
f2 <- glm.synds(smoke ~ sex + age + edu + ls, object = s2, family = "binomial")
summary(f2)
summary(f2, populationInference = TRUE)
```

---

summary.synds *Synthetic data object summaries*

Description

Produce summaries of the synthesised variables for selected copies of synthetic data.

Usage

```
## S3 method for class 'synds'
summary(object, msel = 1, ...)

## 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.
...	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 real data.
msel	index or indices of synthetic data copies for which a summary is produced.
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)),
   visitSequence = (1:ncol(data)), predictorMatrix = NULL,
   m = 1, k = nrow(data), proper = FALSE, nlevelmax = 5,
   maxfaclevels = 60,
   rules = as.list(rep("", ncol(data))),
   rvalues = as.list(rep(NA, ncol(data))),
   contNA = as.list(rep(NA, ncol(data))),
   event = rep(0, ncol(data)),
   smoothing = rep("", ncol(data)),
   denom = rep(0, ncol(data)),
   minbucket = 5,
   drop.not.used = TRUE,
   drop.pred.only = TRUE,
   defaultMethod = c("normrank", "logreg", "polyreg", "polr"),
   diagnostics = FALSE, printFlag = TRUE, ...)

## 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 defaultMethod 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.
visitSequence	a vector of integers of arbitrary length, specifying the column indices of the synthesising/visiting sequence. The default sequence 1:ncol(data) implies that column variables are synthesised/visited from left to right. See details for more information.
predictorMatrix	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 predictorMatrix will have values of 1 in the lower triangle. See details for more information.

m	number of synthetic copies of the real 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 real data (k = n).
proper	a logical value with default set to FALSE. If TRUE proper synthesis is conducted.
nlevelmax	a minimum number of values a numeric variable should have to be treated as numeric. Numeric variables with fewer levels than nlevelmax are changed into factors.
maxfaclevels	a maximum number of factor levels that can be handled by parametric methods.
rules	a vector or a list of rules for restricted values. Restricted values are those that are determined explicitly by values of other variables.
rvalues	a vector or a list of the values corresponding to the rules specified by rules.
contNA	a vector or a list of codes for missing values for continuous variables if different from the R missing data code NA.
event	a vector of integers of length ncol(data) specifying for survival data the column indices for corresponding event indicators. For non-survival data it has a value of 0. Order of variables is exactly the same as in data.
smoothing	a vector of length ncol(data) specifying smoothing method ("density" or "") to be used for each variable in the data. Smoothing can be applied to continuous variables only and the empty method "" should be set for all other variables.
denom	a vector of integers of length ncol(data) specifying for variables to be modelled using binomial regression the column indices for corresponding denominator variables. For other variables it has a value of 0.
minbucket	the minimum number of observations in any terminal node of a CART model.
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.
defaultMethod	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.
printFlag	if TRUE (default) syn will print synthesising history at the console. For silent computation use print = FALSE.
...	additional arguments passed to synthesising functions.
x	an object of class synds; a result of a call to syn.

Details

Only variables that are in `visitSequence` 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 `visitSequence`. All other variables (not in `visitSequence` or in `visitSequence` with a corresponding blank method) can be used as predictors. Including them in `visitSequence` generates a default `predictorMatrix` reflecting the order of variables in the `visitSequence` otherwise `predictorMatrix` has to be adjusted accordingly. All predictors of the variables that are not in `visitSequence` or are in `visitSequence` but with a blank method are removed from `predictorMatrix`.

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 `nlevelmax` 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)

surv.ctree 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

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

Value

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

`call`	the original call to `syn`.
`m`	the number of synthetic versions of the real 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.
`visitSequence`	a vector of column indices of the visiting sequence. The indices refer to the columns in the saved synthesised data.

predictorMatrix
: a matrix specifying the set of predictors used for 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.

smoothing
: a vector specifying smoothing methods applied to each variable 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.

minbucket
: a minimum number of observations in any terminal node of a CART model.

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`.

contNA
: a list of codes for missing values for continuous variables.

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.

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")
rds  <- SD2011[1:2000,vars]

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

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

### multiple synthesis of selected variables with customised methods
s3 <- syn(rds, visitSequence = c(2, 1, 6, 5), m = 2,
```

```
          method = c("logreg","sample","","normrank", "ctree",""))
summary(s3)
summary(s3, msel = 1:2)

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

### handling missing values in continuous variables
contNA.income <- as.list(rep(NA, ncol(rds)))
contNA.income[[4]] <- c(NA,-8)
s5 <- syn(rds, contNA = contNA.income)

### rules for restricted values - marital status of males under 18 should be 'single'
rules.marital <- list("","","age < 18 & sex == 'MALE'","","","")
rvalues.marital <- list(NA,NA,'SINGLE',NA,NA,NA)
s6 <- syn(rds, rules = rules.marital, rvalues = rvalues.marital, method = "parametric")
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, ...)
syn.cart(y, x, xp, smoothing, proper = FALSE, minbucket, 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.

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 stopping rule for the splitting process.

Value

A vector of length k with synthetic values of y.

See Also

syn, syn.surv.ctree, rpart, ctree

---

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.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 methods.
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.

Value

A vector of length k with synthetic values of y.

See Also

syn, syn.norm

---

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, nnet.maxit = 100, nnet.trace = FALSE,
  nnet.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.
`nnet.maxit`	the maximum number of iterations for `nnet`.
`nnet.trace`	switch for tracing optimization for `nnet`.
`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. 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, nnet.maxit = 100, nnet.trace = FALSE,
  nnet.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.
nnet.maxit	the maximum number of iterations for nnet.
nnet.trace	switch for tracing optimization for nnet.
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. 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, proper = FALSE, ...)
```

Arguments

y	an original data vector of length n.
xp	a target length k of a synthetic data vector.
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.

Value

A vector of length k with synthetic values.

See Also

syn

`syn.surv.ctree` *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.surv.ctree(y, yevent, x, xp, proper = FALSE, minbucket, ...)
```

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`

Index