Package ‘synthesizer’

July 10, 2025

Type Package

Title Fast, Robust, and High-Quality Synthetic Data Generation with a Tuneable Privacy-Utility Trade-Off

Version 0.5.0

Maintainer Mark van der Loo <mark.vanderloo@gmail.com>

Description Synthesize numeric, categorical, mixed and time series data. Data circumstances including mixed (or zero-inflated) distributions and missing data patterns are reproduced in the synthetic data. A single parameter allows balancing between high-quality synthetic data that represents correlations of the original data and lower quality but more privacy safe synthetic data without correlations. Tuning can be done per variable or for the whole dataset.

License EUPL

URL https://github.com/markvanderloo/synthesizer

Imports stats

VignetteBuilder simplermarkdown

Depends R (>= 3.5.0)

Suggests tinytest, simplermarkdown

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Mark van der Loo [aut, cre] (ORCID: <https://orcid.org/0000-0002-9807-4686>)

Repository CRAN

Date/Publication 2025-07-10 13:10:05 UTC

Contents

make_synthesizer *Create a function that generates synthetic data*

Description

Create a function that accepts a non-negative integer n, and that returns synthetic data sampled from the emperical (multivariate) distribution of x.

Usage

```
make_synthesizer(x, ...)

## S3 method for class 'numeric'
make_synthesizer(x, na.rm = FALSE, ...)

## S3 method for class 'integer'
make_synthesizer(x, na.rm = FALSE, ...)

## S3 method for class 'logical'
make_synthesizer(x, na.rm = FALSE, ...)

## S3 method for class 'factor'
make_synthesizer(x, na.rm = FALSE, ...)

## S3 method for class 'character'
make_synthesizer(x, na.rm = FALSE, ...)

## S3 method for class 'ts'
make_synthesizer(x, ...)

## S3 method for class 'data.frame'
make_synthesizer(x, na.rm = FALSE, ...)
```

Arguments

`x`	`[vector|data.frame]` Template data to be synthesized.
`...`	arguments passed to other methods
`na.rm`	`[logical]` Remove missing values before creating a synthesizer

Value

A `function` accepting a single integer argument: the number of synthesized values or records to return. For objects of class `ts` `n` must be equal to the length of the original data (this is set as the default).

See Also

Other synthesis: `synthesize()`

Examples

```
synth <- make_synthesizer(cars$speed)
synth(10)


synth <- make_synthesizer(iris)
synth(6)
synth(150)
synth(250)
```

`synthesize` *Create synthetic version of a dataset*

Description

Create `n` values or records based on the emperical (multivariate) distribution of `y`. For data frames it is possible to decorrelate synthetic from the original variables by lowering the value for the `rankcor` parameter.

Usage

```
synthesize(x, na.rm = FALSE, n = NROW(x), rankcor = 1)
```

Arguments

`x`	`[vector|data.frame]` data to synthesize.
`na.rm`	`[logical]` Remove missing values before creating a synthesizer. Set to `TRUE` to avoid synthesizing missing values.
`n`	`[integer]` Number of values or records to synthesize.
`rankcor`	`[numeric]` in $[0, 1]$. Either a single rank correlation value that is applied to all variables, or a vector of the form `c(variable1=ut1lity1,...)`. Variables not explicitly mentioned will have `rankcor=1`. See also the note below. Ignored for all types of `x`, except for objects of class `data.frame`.

Value

A data object of the same type and structure as `x`.

Note

The utility of a synthetic variable is lowered by decorelating the rank correlation between the real and synthetic data. If `rankcor=1`, the synthetic data will ordered such that it has the same rank order as the original data. If `rankcor=0`, no such reordering will take place. For values between 0 and 1, blocks of data are randomly selected and randomly permuted iteratively until the rank correlation between original and synthetic data drops below the parameter.

See Also

Other synthesis: `make_synthesizer()`

Examples

```
synthesize(cars$speed,10)
synthesize(cars)
synthesize(cars,25)

s1 <- synthesize(iris, rankcor=1)
s2 <- synthesize(iris, rankcor=0.5)
s3 <- synthesize(iris, rankcor=c("Species"=0.5))

oldpar <- par(mfrow=c(2,2), pch=16, las=1)
plot(Sepal.Length ~ Sepal.Width, data=iris, col=iris$Species, main="Iris")
plot(Sepal.Length ~ Sepal.Width, data=s1, col=s1$Species, main="Synthetic Iris")
plot(Sepal.Length ~ Sepal.Width, data=s2, col=s2$Species, main="Low utility Iris")
plot(Sepal.Length ~ Sepal.Width, data=s3, col=s3$Species, main="Low utility Species")
par(oldpar)
```

Index