Package ‘synthesizer’

October 11, 2024

Type Package

Title Synthesize Data Based on Empirical Quantile Functions and Rank Order Matching

Version 0.3.1

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

Description Data is synthesized using a combination of inverse transform sampling from the empirical quantile functions for each variable, and then copying the rank order structure from the original dataset. The package also includes a number of functions to measure the utility of synthesized datasets.

License EUPL

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

Imports stats, randomForest

VignetteBuilder simplermarkdown

Depends R (>= 3.5.0)

Suggests tinytest, simplermarkdown

Encoding UTF-8

RoxygenNote 7.3.1

NeedsCompilation no

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

Repository CRAN

Date/Publication 2024-10-11 14:50:02 UTC

Contents

dcor 2
dmean 2
make_synthesizer 3
pmse 5
qa 5
synthesize 6

Index **7**

dcor *Difference between correlation*

Description

Returns the Frobenius norm of the difference between the correlation matrices for numeric columns in synthetic and real data.

Usage

```
dcor(synth, real)
```

Arguments

`synth`	`[data.frame]` Synthetic data
`real`	`[data.frame]` Real data

See Also

Other measures: `dmean()`, `pmse()`, `qa()`

Examples

```
dcor(iris, iris) # 0
dcor(synthesize(cars), cars)
```

dmean *Summarization of location and spread between synthetic and real data*

Description

For each numerical variable in the two datasets, compute the relative difference between the mean (standard deviation) of the real data and the mean (standard deviation) of the synthetic data. The summary is the average of these relative differences over all numerical variables.

Usage

```
dmean(synth, real, tol = 1e-08, ...)

dsd(synth, real, tol = 1e-08, ...)
```

Arguments

`synth`	`[data.frame]` Synthetic data
`real`	`[data.frame]` Real data
`tol`	`[numeric]` Nonnegative tolerance. If the absolute mean (standard deviation) of a variable is smaller than `tol`, it is considered zero. In that case the absolute difference instead of the absolute relative difference is computed.
`...`	Arguments passed to `mean`. e.g. use `trim=c(0.01,0.99)` for mean estimation that is less sensitive to outliers.

Value

`[numeric]` scalar.

Note

Real and synthetic data are expected to have the same column names, orders, and data types.

See Also

Other measures: `dcor()`, `pmse()`, `qa()`

Other measures: `dcor()`, `pmse()`, `qa()`

Examples

```
dmean(cars, cars) # 0
dmean(synthesize(cars), cars)

dsd(cars, cars) # 0
dsd(synthesize(cars), cars)
```

`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 `y`.

Usage

```
make_synthesizer(y)

## S3 method for class 'numeric'
make_synthesizer(y)

## S3 method for class 'integer'
make_synthesizer(y)

## S3 method for class 'logical'
make_synthesizer(y)

## S3 method for class 'factor'
make_synthesizer(y)

## S3 method for class 'character'
make_synthesizer(y)

## S3 method for class 'data.frame'
make_synthesizer(y)
```

Arguments

`y`	`[vector|data.frame]` Template data to be synthesized.

Value

A `function` accepting a single integer argument: the number of synthesized values or records to return.

See Also

Other synthesis: `synthesize()`

Examples

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


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

pmse *Compute the pMSE metric between synthetic and real data*

Description

The propensity mean squared error is defined as $\frac{1}{N}\sum_{i=1}^{N}(p_i - c)^2$, where c is the number of synthetic records, divided by the sum of the number of synthetic and real records.

Usage

```
pmse(synth, real, model = c("lr", "rf"), nrep = NULL)
```

Arguments

`synth`	`[data.frame]` Synthesized data.
`real`	`[real]` Data to compare with the synthesized data.
`model`	`[character]` Model used to compute propensity scores. Options are `"lr"`: logistic regression, and `"rf"`: random forest.
`nrep`	`[integer]` Number of model repetitions to average the pMSE value over. Ignored for `lr`.

Value

`[numeric]` scalar.

See Also

Other measures: `dcor()`, `dmean()`, `qa()`

Examples

```
scars <- synthesize(cars)
pmse(scars, cars)
```

qa *Quality assurance for synthesized data*

Description

Repeatedly synthesize a dataset, record a set of quality measures for each repetition.

Usage

```
qa(real, n = 10)
```

Arguments

`real`	`[data.frame]` A data set to be synthesized.
`n`	`[integer]` Number of repetitions

Value

A `data.frame` with `n` rows and each column a quality measure.

See Also

Other measures: `dcor()`, `dmean()`, `pmse()`

Examples

```
qa(iris)
```

`synthesize` *Create synthetic version of a dataset*

Description

Create `n` values or records based on the emperical (multivariate) distribution of `y`.

Usage

```
synthesize(y, n = NROW(y))
```

Arguments

`y`	`[vector|data.frame]` data to synthesize.
`n`	`[integer]` Number of values or records to synthesize.

Value

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

See Also

Other synthesis: `make_synthesizer()`

Examples

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

Index

∗ **measures**
    dcor, 2
    dmean, 2
    pmse, 5
    qa, 5
∗ **synthesis**
    make_synthesizer, 3
    synthesize, 6

dcor, 2, *3*, *5*, *6*
dmean, *2*, 2, *5*, *6*
dsd *(*dmean*)*, 2

make_synthesizer, 3, *6*

pmse, *2*, *3*, 5, *6*

qa, *2*, *3*, *5*, 5

synthesize, *4*, 6