Package 'sdcTarget'

February 20, 2015

Type Package

Title Statistical Disclosure Control Substitution Matrix Calculator

Author Emmanuel Lazaridis [aut, cre]

Maintainer Emmanuel Lazaridis <emmanuel@lazaridis.eu>

Depends R (>= 2.10.0), methods, magic, foreach, parallel, doParallel, tuple (>= 0.4-02)

Description Classes and methods to calculate and evaluate target matrices for statistical disclosure control.

License CC BY-NC 4.0

Encoding UTF-8

LazyLoad no

URL http://statistics.lazaridis.eu

Version 0.9-11

Date 2014-11-03

NeedsCompilation no

Repository CRAN

Date/Publication 2014-11-03 15:16:24

R topics documented:

`sdcTarget-package` *The sdcTarget Package*

Description

Statistical Disclosure Control Substitution Matrix Calculator.

Provides functions to calculate target matrices for statistical disclosure control.

Details

Package:	sdcTarget
Type:	Package
Version:	0.9-11
Date:	2014-11-03
Depends:	R (>= 2.10.0), methods, magic, foreach, parallel, doParallel, tuple (>= 0.4-02)
Encoding:	UTF-8
License:	CC BY-NC 4.0
LazyLoad:	no
URL:	http://statistics.lazaridis.eu

Classes and methods to calculate and evaluate target matrices for statistical disclosure control.

Parallel processing is now implemented by default to speed up large searches. The number of processors must be specified. If not specified, will default to linear processing.

Author(s)

Emmanuel Lazaridis

References

Summary of talk at RSS 2014 Conference

`estimateTimeToComputeTargetDefinition`
Estimate Time To Complete Calculation of Target Definition

Description

This code provides a rough approximation for the number of seconds that a calculation of target definition may run.

Usage

```
estimateTimeToComputeTargetDefinition(X, cutoff = 1, cores = 1,
  na.recode = TRUE, Hdef = new("sdcHashingDefinitionClass", X = X, na.recode
  = na.recode))
```

Arguments

X	A data frame with the data (not necessarily standardised).
cutoff	The minimum residual cell size.
cores	The number of cores to be used in parallel processing. The default is linear processing: i.e., core = 1.
na.recode	Whether NA values should be treated as levels.
Hdef	A hashing definition (might be generated from a larger set of data).

Examples

```
set.seed(256)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
estimateTimeToComputeTargetDefinition(X = my.X)
```

---

sdcHashClass-class *S4 Hash Class*

Description

This class defines the data hash that is used to identify cells to be targeted for statistical disclosure control.

Usage

```
## S4 method for signature 'sdcHashClass'
initialize(.Object, ...)
```

Arguments

.Object	A sdcHashingDefinitionClass object.
...	The optional parameters specifying the data and other options.

Details

The hashing definition presently handles only categorical fields, which limits the applicability of this software accordingly.

Does not currently permit use of an arbitrary hashing definition.

Methods (by generic)

- initialize:

Slots

`.Data` A matrix.

`Hdef` The hashing definition on which the `sdcHashClass` object is built.

`forward` Logical indicating whether a forward hash or a backward hash, in terms of field order, has been used.

`hash` A character string containing one hash representation for each data record.

Examples

```
new("sdcHashClass")
new("sdcHashClass")
my.X <- data.frame(matrix(ifelse(runif(5000)>.5, TRUE, FALSE), ncol = 50))
new("sdcHashClass", X = my.X)
new("sdcHashClass", X = my.X, na.recode = FALSE, which = 2:4,
    forwardHashing = TRUE)
```

---

sdcHashingDefinitionClass-class

Hashing Definition (S4 Class)

Description

This class defines the data hash that is used to identify cells to be targeted for statistical disclosure control.

Usage

```
## S4 method for signature 'sdcHashingDefinitionClass'
initialize(.Object, ...)
```

Arguments

`.Object`	A `sdcHashingDefinitionClass` object.
`...`	The optional parameters specifying the data and whether `NA` values are to be recoded and hashed. Recoding is `TRUE` by default.

Details

The hashing definition presently handles only categorical fields, which limits the applicability of this software accordingly.

Methods (by generic)

- `initialize`:

Slots

`parts` The number of hashing components.

`coverage` The number of elements in each hashing component.

`lengths` The number of levels of each field to be hashed.

`na.exists` Whether there is an NA value in each field to be hashed.

`na.recode` Whether NA values should be treated as levels.

`levels` The number of levels of each field to be hashed.

Examples

```
new("sdcHashingDefinitionClass")
new("sdcHashingDefinitionClass")
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
new("sdcHashingDefinitionClass", X = my.X)
new("sdcHashingDefinitionClass", X = my.X, na.recode = FALSE)
```

---

sdcProcessCells *Identify or Remove Small or Large Cells From a Hash List*

Description

Returns a hash list identifying the appropriate cells, or a subset of the data provided with the specified cells removed.

Usage

```
identifySmallCells(Y, cutoff = 1)

identifyBigCells(Y, cutoff = 1)

identifySmallCellRecords(Y, cutoff = 1)

identifyBigCellRecords(Y, cutoff = 1)

removeSmallCellRecords(X, Y, cutoff = 1)

removeBigCellRecords(X, Y, cutoff = 1)
```

Arguments

`Y`	A hash list as returned by `sdcHashClass`.
`cutoff`	The lower or upper cutoff of cell size.
`X`	A data frame corresponding to `Y`.

See Also

sdcHashClass,

Examples

```
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.hc <- new("sdcHashClass", X = my.X)
identifySmallCells(my.hc)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.hc <- new("sdcHashClass", X = my.X)
identifyBigCells(my.hc)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.hc <- new("sdcHashClass", X = my.X)
identifySmallCellRecords(my.hc)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.hc <- new("sdcHashClass", X = my.X)
identifyBigCellRecords(my.hc)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.hc <- new("sdcHashClass", X = my.X)
removeSmallCellRecords(my.X, my.hc)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.hc <- new("sdcHashClass", X = my.X)
removeBigCellRecords(my.X, my.hc)
```

---

sdcSubstitutionMatrixClass-class

S4 Substitution Matrix

Description

The substitution matrix is a matrix with the same dimensions as the data from which it is derived, that indicates which elements are to be subjected to a statistical disclosure control process.

Usage

```
## S4 method for signature 'sdcSubstitutionMatrixClass'
initialize(.Object, ...)
```

Arguments

.Object	An sdcSubstitutionMatrixClass object.
...	The optional parameters specifying the basis of the substitution matrix. These include an indicator of forwards or backwards matrix calculation (forwards), a cutoff value relative to entries in the target matrix (cutoff), and the target matrix basis for the calculation (T).

Details

The substitution matrix is calculated from a target matrix. Specification of forwards direction with a `cutoff` of 0 results in a complete data synthesis substitution matrix. Specification of forwards direction with a `cutoff` of 1, or backwards direction with a sufficiently large `cutoff`, results in the maximum partial synthesis substitution matrix. Specification of backwards direction with a `cutoff` of 1 results in the minimum partial synthesis substitution matrix. Maximum partial synthesis is the default.

Methods (by generic)

- `initialize`:

Slots

`.Data` A matrix.

`forwards` Indicates the direction in which to process the target matrix. Defaults to `TRUE`.

`cutoff` Cutoff with respect to which an element in the target matrix should be indicated for substitution. Default value is 1.

`T` A target matrix object.

See Also

sdcTargetMatrixClass,

Examples

```
set.seed(256)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.smc <- new("sdcSubstitutionMatrixClass",
              T = new("sdcTargetMatrixClass",
                      Tdef = new("sdcTargetDefinitionClass", X = my.X)))
```

---

sdcTargetDefinitionClass-class

S4 Target Definition

Description

The SDC target definition is used to calculate a target matrix.

Usage

```
## S4 method for signature 'sdcTargetDefinitionClass'
initialize(.Object, ...)
```

Arguments

.Object	An sdcTargetDefinitionClass object.
...	The optional parameters specifying the data and other options. Among these are

Name	Description
X	A data frame with the data, not necessarily standardised.
Hdef	A hashing definition, if generated externally, perhaps because it comes from a larger set of data.
cutoff	The minimum residual cell size.

Details

The hashing definition presently handles only categorical fields, which limits the applicability of this software accordingly.

Methods (by generic)

- `initialize`:

Slots

`dim` The dimensions of the data on which the target definition is calculated.

`dimnames` A list of row and column names of the data on which the target definition is calculated, in the form returned by `dim`.

`cutoff` The minimum residual cell size.

`rownames` The row names of the target elements.

`size` The size of each target element.

`index` The index of each target element.

`hash` The target hash vector.

See Also

`sdcTargetMatrixClass`,

Examples

```
new("sdcTargetDefinitionClass")
set.seed(256)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
new("sdcTargetDefinitionClass", X = my.X)
```

sdcTargetMatrixClass-class
S4 Target Matrix

Description

The taget matrix is a matrix with the same dimensions as the data from which it is derived, that indicates the number of combinations at a specific level of targetting for which the synthesis of a data element will make the record "sufficiently common".

Usage

```
## S4 method for signature 'sdcTargetMatrixClass'
initialize(.Object, ...)
```

Arguments

.Object	An sdcTargetMatrixClass object.
...	The optional parameters specifying the basis of the target matrix (Tdef).

Details

Additional information is stored in the slots.

Methods (by generic)

- initialize:

Slots

.Data A matrix.

Tdef A target definition class object.

See Also

sdcTargetDefinitionClass,

Examples

```
set.seed(256)
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.tdef <- new("sdcTargetDefinitionClass", X = my.X)
new("sdcTargetMatrixClass", Tdef = my.tdef)
```

show,sdcHashClass-method
Show Method For sdcHashClass *Objects*

Description

Returns an abbreviated summary of a sdcHashClass object.

The summary method runs and returns the output of some calculations on an sdcHashClass object, together with an abbreviated summary of the object itself.

Usage

```
## S4 method for signature 'sdcHashClass'
show(object)

## S4 method for signature 'sdcHashClass'
summary(object)
```

Arguments

object	A sdcHashClass object.

Details

As described in the documentation for show, the default for printing out S4 objects is a call to a show method, whereas print is the default for S3 objects. This gives the possibility to have an alternative way to print the contents of a user-defined S4 class object.

Methods (by class)

- sdcHashClass:
- sdcHashClass:

Examples

```
show(new("sdcHashClass"))
summary(new("sdcHashClass"))
my.X <- data.frame(matrix(ifelse(runif(500)>.5, TRUE, FALSE), ncol = 5))
my.hc <- new("sdcHashClass", X = my.X)
summary(my.hc)
```

`[,sdcHashClass-method` *Get a Hash List Subset*

Description

Returns a subset of a `sdcHashClass` object.

Usage

```
## S4 method for signature 'sdcHashClass'
x[i]
```

Arguments

`x`	An `sdcHashClass` object.
`i`	An index of the hash.

Details

Does what it says on the package.

See Also

`sdcHashClass`,

Examples

```
my.X <- data.frame(matrix(ifelse(runif(5000)>.5, TRUE, FALSE), ncol = 50))
my.hc <- new("sdcHashClass", X = my.X)
my.hc[2:3]
```

Index