Package: blockrand2 (via r-universe)

August 22, 2024

Type Package

Title Block Randomisation for Clinical Studies

Version 0.1.0

Author Hauke Sonnenberg

Maintainer Hauke Sonnenberg <hauke.sonnenberg@gmx.de>

Description Create block randomised treatment lists for clinical studies.

Imports rmarkdown, knitr

License GPL (>= 2)

LazyData TRUE

RoxygenNote 5.0.1

Repository https://hsonne.r-universe.dev

RemoteUrl https://github.com/hsonne/blockrand2

RemoteRef HEAD

RemoteSha 96fd8291d20a0e5ff2c27730adc54681ba04897f

Contents

blocksize *Get a suitable blocksize for a given number of subjects*

Description

Get a suitable blocksize for a given number of subjects

Usage

```
blocksize(x, N)
```

Arguments

x	number of patients in stratum
N	number of treatments

Value

integer vector of length one representing the blcksize

Examples

```
# Blocksize for 10 subjects and 2 treatments
blocksize(10, 2)

# Blocksize for 10 subjects and 3 treatments
blocksize(10, 3)
```

createCombinations *Create all possible combinations of values*

Description

Create all possible combinations of values given in a list of value vectors

Usage

```
createCombinations(x)
```

Arguments

x	List of vectors containing the values to be combined. The names of the list elements will appear as column names in the output data frame.

Value

Data frame with each row representing a possible combination of values.

Examples

```
createCombinations(list(
  sex = c("male", "female"),
  smokes = c("yes", "sometimes", "no"),
  cancer = c("yes", "no")
))
```

`createRandomisationDoc` *Create block randomisation list documents*

Description

Create block randomisation list documents in three possible formats: HTML, PDF (requires LaTeX) and Microsoft Word

Usage

```
createRandomisationDoc(patients, strataVars, treatments, header = list(title =
  "Randomisierungsliste zur Studie: title", author = "author", date =
  Sys.Date()), format = c("html", "word", "pdf"), newpage = FALSE,
  file.rmd = system.file("extdata", "randomList.Rmd", package = "blockrand2"),
  outdir = file.path(tempdir(), "blockrand2"))
```

Arguments

`patients`	data frame containing the patient's data as returned by `createTestdata`
`strataVars`	List of strata variables, each of which is defined in the form of a vector of possible (character) values.
`treatments`	named vector of character defining the possible treatments. The names are used as acronyms, the values as the full treatment names
`header`	list with elements `title`, `author`, `date` defining the values to be used in the title, author and date field, respectively, in the header of the document
`format`	vector of elements from `c("html", "word", "pdf")` defining the types of documents to be created
`newpage`	if `TRUE` and `"pdf"` is in `format` each stratum will appear on its own page in the created PDF document
`file.rmd`	full path to the RMarkdown file that is rendered to create the randomisation list documents. By default the file `randomList.Rmd` in the subfolder `extdata` of the package installation directory (returned by `system.path(package = "blockrand2")` is used. You may want to make a copy of this file, modify it and set `file.rmd` to the path of that new file to override the appearence of the created output.
`outdir`	full path to the output directory to which the created document files are written. By default the files are written to a subfolder `blockrand2` in the current R session's temporary directory

Value

list containing the full paths to the created files as values and the file extensions (`html`, `docx`, `pdf`) as element names

See Also

https://github.com/hsonne/blockrand2 (provides a short tutorial in the README file)

Examples

```
## Not run:

# Define title of the study, the author and the date
header <- list(
  title = "Randomisation lists for the study: *Jokes against craziness*",
  author = "Hauke Sonnenberg",
  date = "2016-08-27"
)

# Define stratum variables
strataVars <- list(
  sex = c("male", "female"),
  crazyness = c("weak", "medium", "strong")
)

# Define the treatments
treatments <- c(
  joke = "Tell funny jokes",
  nojoke = "Keep serious!"
)

# Create some patient's testdata using the stratum levels
patients <- createTestdata(strataVars, 40)

# Create the randomisation list documents. We store the paths to the created
# files that are returned invisibly by createRandomisationDoc() in a variable
files <- createRandomisationDoc(
  patients = patients,
  strataVars = strataVars,
  treatments = treatments,
  header = header,
  newpage = TRUE
)

# Show the paths of the created files
files

# Open the html file in the default browser
browseURL(files$html)

# Open the pdf file in the default PDF viewer
system(paste(getOption("pdfviewer"), files$pdf))
```

```
## End(Not run)
```

`createRandomSequences` *Create random sequences of blockwise equally distributed values*

Description

Create random sequences of blockwise equally distributed values. The blocksizes are chosen according to the number of values to be created.

Usage

```
createRandomSequences(x, counts, names.strata = NULL)
```

Arguments

`x`	vector of values to be chosen from
`counts`	vector of integers defining the lengths of vectors of values to be created
`names.strata`	vector of character to be used as element names in the output list

Value

list of vectors of values out of `x` with lengths according to the values given in `counts`

Examples

```
# Define stratum levels
strata <- list(
 sex = c("male", "female"),
 medication = c("yes", "no"),
 lesion = c("low", "high")
)

# Create some testdata using the stratum levels and group and count by stratum
byStratum <- stratify(createTestdata(strata, 50))

# Create random sequences of treatments (A, B)
sequences <- createRandomSequences(
 x = c("A", "B"),
 counts = byStratum$n,
 names.strata = byStratum$stratum
)

# Append a column "treatments" showing the sequences of treatments as comma
# separated lists
byStratum$treatments <- sapply(sequences, paste, collapse = ",")

# Show the result
byStratum
```

createTestdata *Create testdata for clinical studies*

Description

Create testdata for clinical studies

Usage

```
createTestdata(strataVars, n = if (is.null(n.per.stratum)) 30 else NULL,
  n.per.stratum = NULL, format.patient = "P%03d", offset.patient = 0)
```

Arguments

strataVars	List of strata variables, each of which is defined in the form of a vector of possible (character) values.
n	Number of records to be created
n.per.stratum	optional. Vector of integer numbers defining the number of records for each possible stratum. The length of this vector must be equal to the number of possible strata (resulting from the product of lengths of vectors in strataVars)
format.patient	Passed as argument format to sprintf() when creating a unique id for each patient
offset.patient	integer number to be added to the generated patient numbers 1:n. Defaults to 0.

Value

Data frame with a column patient containing the patient's ID and one column for each stratum, named according to the element names in strataVars

Examples

```
strataVars <- list(
 sex = c("female", "male"),
 medication = c("yes", "no")
)

createTestdata(strataVars, n = 20)

# Define the number of patients in each stratum (there are four possible
# strata: length(strataVars$sex) * length(strataVars$medication))
createTestdata(strataVars, n.per.stratum = c(5, 6, 3, 4))
```

`equalDistribution` *Create a random sequence of equally distributed values*

Description

Create a random sequence of equally distributed values

Usage

```
equalDistribution(x, length.out = length(x), do.stop = TRUE)
```

Arguments

`x`	Vector of values to be chosen from
`length.out`	Length of the output vector to be created
`do.stop`	If `TRUE` (default) an error is thrown if `length.out` is not a multiple of the length of `x`. Otherwise no error is thrown.

Value

Vector of `length.out` values all of which are elements of `x`. If `length.out` is a multiple of the lenght of `x` the values are equally distributed. Otherwise (and if `do.stop` is `FALSE` so that no error is thrown) the frequencies of the values differ by one at most.

Examples

```
y1 <- equalDistribution(x = LETTERS[1:2], 10)

# Check the distribution with table
table(y1)

# Do all values occur with the same frequency?
all(diff(table(y1)) == 0)

y2 <- equalDistribution(x = LETTERS[1:3], length.out = 11, do.stop = FALSE)

# Do the frequencies differ by one at most?
all(diff(table(y2)) <= 1)
```

`stratify` *Group and count data by combinations of strata variables*

Description

Group data by combinations of stratum values and count the records falling into each combination.

Usage

```
stratify(x, strataVars = toStrataVars(x, exclude = names(x)[1]),
  column.n = "n", column.stratum = "stratum", format.stratum = "S%02d")
```

Arguments

`x`	Data frame with stratum values in columns
`strataVars`	List of strata variables. By default it is created from all but the first columns of `x`.
`column.n`	Column name to be used in the output for the column containing the number of records falling into the combination of stratum values.
`column.stratum`	Column name to be used in the output for the column containing the stratum identifier
`format.stratum`	format string to be used in `format` to create a unique id for each stratum

Value

Data frame with each row representing a combination of stratum values

Examples

```
# Define stratifying variables
strataVars <- list(sex = c("female", "male"), medication = c("yes", "no"))

# Create some random testdata
x <- createTestdata(strataVars)

# Count records in each stratum
stratify(x, strataVars)
stratify(x, strataVars, column.n = "n.patients")

# Create testdata with defined numbers of records in each stratum
n.per.stratum <- 1:4
y <- createTestdata(strataVars, n.per.stratum = n.per.stratum)

# Count records in each stratum
(counts <- stratify(y, strataVars))

# Check if the numbers of records in each stratum are as expected
all(counts$n == n.per.stratum)
```

toStratumString *Create a string describing the stratum from a one-row data frame*

Description

Create a string describing the stratum from a one-row data frame

Usage

```
toStratumString(stratumInfo)
```

Arguments

stratumInfo	data frame with one row and stratum information in all but the very first (stratum) and the very last (n) column.

Value

character string of the form attribute1 = value1, attribute2 = value2, etc.

Examples

```
stratumInfo <- data.frame(
 stratum = "S01",
 sex = "male",
 pretreatment = "no",
 n = 5
)

toStratumString(stratumInfo)
```

Index