Package ‘experDesign’

April 22, 2021

Title Design Experiments for Batches

Version 0.1.0

Description Distributes samples in batches while making batches homogeneous according to their description. Allows for an arbitrary number of variables, both numeric and categorical. For quality control it provides functions to subset a representative sample.

License MIT + file LICENSE


BugReports https://github.com/llrs/experDesign/issues

Depends R (>= 3.5.0)

Imports methods, stats

Suggests knitr, rmarkdown, testthat (>= 2.1.0), covr, MASS, spelling

biocViews

VignetteBuilder knitr

Encoding UTF-8

Language en-US

LazyData false

RoxygenNote 7.1.1

NeedsCompilation no

Author Lluís Revilla Sancho [aut, cre]
  (<https://orcid.org/0000-0001-9747-2570>)

Maintainer Lluís Revilla Sancho <lluis.revilla@gmail.com>

Repository CRAN

Date/Publication 2021-04-22 14:10:06 UTC
R topics documented:

- experDesign-package .................................................. 2
- batch_names .................................................................. 3
- check_index ................................................................... 3
- create_subset ............................................................... 4
- design ............................................................................ 5
- distribution ................................................................... 6
- entropy ........................................................................... 6
- evaluate_entropy ............................................................ 7
- evaluate_independence .................................................... 8
- evaluate_index ............................................................... 9
- evaluate_mad ................................................................... 9
- evaluate_mean ............................................................... 10
- evaluate_na ..................................................................... 11
- evaluate_orig ............................................................... 12
- evaluate_sd ..................................................................... 12
- extreme_cases ............................................................. 13
- inspect ........................................................................... 14
- optimum ......................................................................... 15
- qcSubset ......................................................................... 16
- replicates ......................................................................... 16
- spatial ............................................................................ 17
- use_index ......................................................................... 18

Index 19

---

experDesign-package  experDesign: Expert experiment design in batches

Description

Enables easy distribution of samples per batch avoiding batch and confounding effects by randomization of the variables in each batch.

Details

The most important function is `design()`, which distributes samples in batches according to the information provided.

To help in the bench there is the `inspect()` function that appends the group to the data provided.

Author(s)

Lluís Revilla
**batch_names**

*Name the batch*

**Description**

Given an index return the name of the batches the samples are in.

**Usage**

```r
batch_names(i)
```

**Arguments**

- **i**
  
  A list of numeric indices.

**Value**

A character vector with the names of the batch for each the index.

**See Also**

- `create_subset()` for the inverse look at `use_index()`

**Examples**

```r
index <- create_subset(100, 50, 2)
batch <- batch_names(index)
head(batch)
```

---

**check_index**

*Check index distribution on batches*

**Description**

Report the statistics for each subset and variable compared to the original.

**Usage**

```r
check_index(pheno, index, omit = NULL)
```

**Arguments**

- **pheno**
  
  Data.frame with the sample information.

- **index**
  
  A list of indices indicating which samples go to which subset.

- **omit**
  
  Name of the columns of the pheno that will be omitted.
Value

A matrix with the differences with the original data.

See Also

Functions that create an index `design()`, `replicates()`, `spatial()`. See also `create_subset()` for a random index.

Examples

```r
index <- create_subset(50, 24)
metadata <- expand.grid(height = seq(60, 80, 5), weight = seq(100, 300, 50),
                       sex = c("Male","Female"))
check_index(metadata, index)
```

---

`create_subset`  
*Create index of subsets of a data*

Description

Index of the samples grouped by batches.

Usage

```r
create_subset(size_data, size_subset = NULL, n = NULL, name = "SubSet")
```

Arguments

- `size_data`  
  A numeric value of the amount of samples to distribute.
- `size_subset`  
  A numeric value with the amount of samples per batch.
- `n`  
  A numeric value with the number of batches.
- `name`  
  A character used to name the subsets, either a single one or a vector the same size as `n`.

Value

A random list of indices of the samples.

See Also

`batch_names()`, `use_index()` if you already have a factor to be used as index.

Examples

```r
index <- create_subset(100, 50, 2)
```
design

Design a batch experiment

Description

Given some samples it distribute them in several batches, trying to have equal number of samples per batch. It can handle both numeric and categorical data.

Usage

```r
design(pheno, size_subset, omit = NULL, iterations = 500, name = "SubSet")
```

Arguments

- **pheno**: Data.frame with the sample information.
- **size_subset**: Numeric value of the number of sample per batch.
- **omit**: Name of the columns of the pheno that will be omitted.
- **iterations**: Numeric value of iterations that will be performed.
- **name**: A character used to name the subsets, either a single one or a vector the same size as n.

Value

The indices of which samples go with which batch.

See Also

The `evaluate_*` functions and `create_subset()`.

Examples

```r
data(survey, package = "MASS")
index <- design(survey[, c("Sex", "Smoke", "Age")], size_subset = 50, 
               iterations = 50)
index
```
distribution  

Description
Checks if all the values are maximally distributed in the several batches. Aimed for categorical variables.

Usage
distribution(report, column)

Arguments
- **report**  A data.frame which must contain a batch column. Which can be obtained with `inspect()`.
- **column**  The name of the column one wants to inspect.

Value
TRUE if the values are maximal distributed, otherwise FALSE.

Examples
data(survey, package = "MASS")
columns <- c("Sex", "Age", "Smoke")
index <- design(pheno = survey[, columns], size_subset = 70, iterations = 10)
batches <- inspect(index, survey[, columns])
distribution(batches, "Sex")
distribution(batches, "Smoke")

entropy  

Description
Calculates the entropy of a category. It uses the amount of categories to scale between 0 and 1.

Usage
entropy(x)

Arguments
- **x**  A character or vector with two or more categories
evaluate_entropy

Value

The numeric value of the Shannon entropy scaled between 0 and 1.

Note

It omits the NA if present.

Examples

entropy(c("H", "T", "H", "T"))
entropy(c("H", "T", "H", "T", "H", "H", "H"))
entropy(c("H", "H", "H", "H", "H", "H", NA))

Description

Looks if the nominal or character columns are equally distributed according to the entropy and taking into account the independence between batches. If any column is different in each row it is assumed to be the sample names and thus omitted.

Usage

evaluate_entropy(i, pheno)

Arguments

i list of numeric indices of the data.frame
pheno Data.frame with information about the samples

Value

Value to minimize

See Also

Other functions to evaluate samples: evaluate_independence(), evaluate_index(), evaluate_mad(), evaluate_mean(), evaluate_na(), evaluate_orig(), evaluate_sd()
Other functions to evaluate categories: evaluate_independence(), evaluate_na()
Examples

data(survey, package = "MASS")
index <- design(survey[, c("Sex", "Smoke", "Age")], size_subset = 50,
iterations = 50)
# Note that numeric columns will be omitted:
evaluate_entropy(index, survey[, c("Sex", "Smoke", "Age")])
evaluate_independence(index, survey[, c("Sex", "Smoke", "Age")])

evaluate_independence  
*Compare independence by chisq.test*

Description

Looks the independence between the categories and the batches.

Usage

evaluate_independence(i, pheno)

Arguments

- `i`  
Index of subsets.

- `pheno`  
A data.frame with the information about the samples.

Value

Returns a vector with the p-values of the chisq.test between the category and the subset.

See Also

Other functions to evaluate samples: `evaluate_entropy()`, `evaluate_index()`, `evaluate_mad()`, `evaluate_mean()`, `evaluate_na()`, `evaluate_orig()`, `evaluate_sd()`

Other functions to evaluate categories: `evaluate_entropy()`, `evaluate_na()`

Examples

data(survey, package = "MASS")
index <- design(survey[, c("Sex", "Smoke", "Age")], size_subset = 50,
iterations = 50)
# Note that numeric columns will be omitted:
evaluate_independence(index, survey[, c("Sex", "Smoke", "Age")])
**evaluate_index**

*Evaluates a data.frame*

**Description**
Measures several indicators per group

**Usage**
```r
evaluate_index(i, pheno)
```

**Arguments**
- `i` Index
- `pheno` Data.frame with information about the samples

**Value**
An array of three dimensions with the mean, standard deviation (`sd()`) and median absolute deviation (`mad()`) of the numeric variables, the entropy of the categorical and the number of NA by each subgroup.

**See Also**
- If you have already an index you can use `use_index()`.
- Other functions to evaluate samples: `evaluate_entropy()`, `evaluate_independence()`, `evaluate_mad()`, `evaluate_mean()`, `evaluate_na()`, `evaluate_orig()`, `evaluate_sd()`

**Examples**
```r
data(survey, package = "MASS")
index <- create_subset(nrow(survey), 50, 5)
ev_index <- evaluate_index(index, survey[, c("Sex", "Smoke")])
ev_index["entropy", , ]
```

---

**evaluate_mad**

*Evaluate median absolute deviation*

**Description**
Looks for the median absolute deviation values in each subgroup.

**Usage**
```r
evaluate_mad(i, pheno)
```
evaluate_mean

Arguments

i       List of indices
pheno   Data.frame with information about the samples

Value

A vector with the mean difference between the median absolute deviation of each group and the original mad.

See Also

Other functions to evaluate samples: evaluate_entropy(), evaluate_independence(), evaluate_index(), evaluate_mean(), evaluate_na(), evaluate_orig(), evaluate_sd()
Other functions to evaluate numbers: evaluate_mean(), evaluate_na(), evaluate_sd()

Examples

data(survey, package = "MASS")
index <- design(survey[, c("Sex", "Smoke", "Age")], size_subset = 50,
iterations = 50)
# Note that categorical columns will be omitted:
evaluate_mad(index, survey[, c("Sex", "Smoke", "Age")])

evaluate_mean   Evaluates the mean of the numeric values

Description

Looks for the mean of the numeric values

Usage

evaluate_mean(i, pheno)

Arguments

i       List of indices
pheno   Data.frame with information about the samples

Value

A matrix with the mean value for each column for each subset

See Also

Other functions to evaluate samples: evaluate_entropy(), evaluate_independence(), evaluate_index(), evaluate_mad(), evaluate_na(), evaluate_orig(), evaluate_sd()
Other functions to evaluate numbers: evaluate_mad(), evaluate_na(), evaluate_sd()
evaluate_na

Evaluate the dispersion of NAs

Description

Looks how are NA distributed in each subset

Usage

evaluate_na(i, pheno)

Arguments

i  list of numeric indices of the data.frame

pheno  Data.frame

Value

The optimum value to reduce

See Also

Other functions to evaluate samples: evaluate_entropy(), evaluate_independence(), evaluate_index(), evaluate_mad(), evaluate_mean(), evaluate_orig(), evaluate_sd()

Other functions to evaluate categories: evaluate_entropy(), evaluate_independence()

Other functions to evaluate numbers: evaluate_mad(), evaluate_mean(), evaluate_sd()

Examples

samples <- 10
m <- matrix(rnorm(samples), nrow = samples)
m[sample(seq_len(samples), size = 5), ] <- NA # Some NA
i <- create_subset(samples, 3, 4) # random subsets
evaluate_na(i, m)
evaluate_orig

Evaluate each variable provided

Description
Measure some summary statistics of the whole cohort of samples

Usage
evaluate_orig(phaeno)

Arguments
phaeno Data.frame with information about the samples

Value
A matrix with the mean, standard deviation, MAD values of the numeric variables, the entropy of the categorical, and the amount of NA per variable.

See Also
Other functions to evaluate samples: evaluate_entropy(), evaluate_independence(), evaluate_index(), evaluate_mad(), evaluate_mean(), evaluate_na(), evaluate_sd()

Examples
data(survey, package = "MASS")
evaluate_orig(survey[, c("Sex", "Age", "Smoke")])

evaluate_sd

Evaluates the mean of the numeric values

Description
Looks for the standard deviation of the numeric values

Usage
evaluate_sd(i, pheno)

Arguments
i List of indices
pheno Data.frame with the samples
extreme_cases

Value
A matrix with the standard deviation value for each column for each subset

See Also
Other functions to evaluate samples: evaluate_entropy(), evaluate_independence(), evaluate_index(), evaluate_mad(), evaluate_mean(), evaluate_na(), evaluate_orig()
Other functions to evaluate numbers: evaluate_mad(), evaluate_mean(), evaluate_na()

Examples

data(survey, package = "MASS")
index <- design(survey[, c("Sex", "Smoke", "Age")], size_subset = 50, iterations = 50)
# Note that categorical columns will be omitted:
evaluate_sd(index, survey[, c("Sex", "Smoke", "Age")])

Description
Subset some samples that are mostly different.

Usage

extreme_cases(phen, size, omit = NULL, iterations = 500)

Arguments

phen Data.frame with the sample information.
size The number of samples to subset.
omit Name of the columns of the pheno that will be omitted.
iterations Numeric value of iterations that will be performed.

Value
A vector with the number of the rows that are selected.

See Also

optimum()
Examples

```r
metadata <- expand.grid(height = seq(60, 80, 5), weight = seq(100, 300, 50),
                        sex = c("Male", "Female"))
sel <- extreme_cases(metadata, 10)
# We can see that it selected both Female and Males and wide range of height
# and weight:
metadata[sel, ]
```

---

**inspect**

*Inspect the index*

**Description**

Given the index and the data of the samples append the batch assignment

**Usage**

```r
inspect(i, pheno, omit = NULL, index_name = "batch")
```

**Arguments**

- `i`: List of indices of samples per batch
- `pheno`: Data.frame with the sample information.
- `omit`: Name of the columns of the `pheno` that will be omitted.
- `index_name`: Column name of the index of the resulting data.frame.

**Value**

The data.frame with a new column `batch` with the name of the batch the sample goes to.

**Examples**

```r
data(survey, package = "MASS")
columns <- c("Sex", "Age", "Smoke")
index <- design(pheno = survey[, columns], size_subset = 70,
                 iterations = 10)
batches <- inspect(index, survey[, columns])
head(batches)
```
optimum

Optimum values for batches

Description

Calculates the optimum values for number of batches or size of the batches. If you need to do several batches it can be better to distribute it evenly and add replicates.

Usage

optimum_batches(size_data, size_subset)
optimum_subset(size_data, batches)
sizes_batches(size_data, size_subset, batches)

Arguments

- size_data: A numeric value of the number of samples to use.
- size_subset: Numeric value of the number of sample per batch.
- batches: A numeric value of the number of batches.

Value

optimum_batches: A numeric value with the number of batches to use.

optimum_subset: A numeric value with the maximum number of samples per batch of the data.

sizes_batches: A numeric vector with the number of samples in each batch.

Examples

```r
size_data <- 50
size_batch <- 24
(batches <- optimum_batches(size_data, size_batch))
# So now the best number of samples for each batch is less than the available
(size <- optimum_subset(size_data, batches))
# The distribution of samples per batch
sizes_batches(size_data, size, batches)
```
**qcSubset**

*Random subset*

**Description**

Select randomly some samples from an index

**Usage**

\[
\text{qcSubset(index, size, each = FALSE)}
\]

**Arguments**

- **index**: A list of indices indicating which samples go to which subset.
- **size**: The number of samples that should be taken.
- **each**: A logical value if the subset should be taken from all the samples or for each batch.

**Examples**

```r
set.seed(50)
index <- create_subset(100, 50, 2)
QC_samples <- qcSubset(index, 10)
QC_samplesBatch <- qcSubset(index, 10, TRUE)
```

**replicates**

*Design a batch experiment with experimental controls*

**Description**

To ensure that the batches are comparable some samples are processed in each batch. This function allows to take into account that effect. It uses the most different samples as controls as defined with `extreme_cases()`.

**Usage**

\[
\text{replicates(pheno, size_subset, controls, omit = NULL, iterations = 500)}
\]

**Arguments**

- **pheno**: Data.frame with the sample information.
- **size_subset**: Numeric value of the number of sample per batch.
- **controls**: The numeric value of the amount of technical controls per batch.
- **omit**: Name of the columns of the pheno that will be omitted.
- **iterations**: Numeric value of iterations that will be performed.
spatial

Value
A index with some samples duplicated in the batches

See Also
design(), extreme_cases().

Examples
```r
samples <- data.frame(L = letters[1:25], Age = rnorm(25))
index <- replicates(samples, 5, controls = 2, iterations = 10)
head(index)
```

spatial

Distribute the sample on the plate

Description
This function assumes that to process the batch the samples are distributes in a plate. Sometimes you know in advance the

Usage
```r
spatial(
  index,
  pheno,
  omit = NULL,
  remove_positions = NULL,
  rows = LETTERS[1:5],
  columns = 1:10,
  iterations = 500
)
```

Arguments
```
  index          A list with the samples on each subgroup, as provided from design() or replicates().
  pheno          Data.frame with the sample information.
  omit           Name of the columns of the pheno that will be omitted.
  remove_positions
                 Character, name of positions.
  rows           Character, name of the rows to be used.
  columns        Character, name of the rows to be used.
  iterations     Numeric value of iterations that will be performed.
```

Value
The indices of which samples go with which batch.
Examples

```r
data(survey, package = "MASS")
index <- design(survey[, c("Sex", "Smoke", "Age")], size_subset = 50,
               iterations = 25)
index2 <- spatial(index, survey[, c("Sex", "Smoke", "Age")], iterations = 25)
head(index2)
```

---

**use_index**

Convert a factor to index

### Description

Convert a given factor to an accepted index

### Usage

```r
use_index(x)
```

### Arguments

- `x` A character or a factor to be used as index

### See Also

You can use `evaluate_index()` to evaluate how good an index is. For the inverse look at `batch_names()`.

### Examples

```r
use_index(plates)
```
Index

* functions to evaluate categories
  evaluate_entropy, 7
  evaluate_independence, 8
  evaluate_na, 11
* functions to evaluate numbers
  evaluate_mad, 9
  evaluate_mean, 10
  evaluate_na, 11
  evaluate_sd, 12
* functions to evaluate samples
  evaluate_entropy, 7
  evaluate_independence, 8
  evaluate_index, 9
  evaluate_mad, 9
  evaluate_mean, 10
  evaluate_na, 11
  evaluate_orig, 12
  evaluate_sd, 12

batch_names, 3
batch_names(), 4, 18

check_index, 3
create_subset, 4
create_subset(), 3–5

design, 5
design(), 2, 4, 17
distribution, 6

design, 5
design(), 2, 4, 17
distribution, 6
design, 5
design(), 2, 4, 17
distribution, 6

test

entropy, 6
evaluate_entropy, 7, 8–13
evaluate_independence, 7, 8, 9–13
evaluate_index, 7, 8, 9, 10–13
evaluate_index(), 18
evaluate_mad, 7–9, 9, 10–13
evaluate_mean, 7–10, 10, 11–13
evaluate_na, 7–10, 11, 12, 13
evaluate_orig, 7–11, 12, 13
evaluate_sd, 7–12, 12

experDesign (experDesign-package), 2
experDesign-package, 2
extreme_cases, 13
extreme_cases(), 16, 17

inspect, 14
inspect(), 2, 6

mad(), 9

optimum, 15
optimum(), 13
optimum_batches (optimum), 15
optimum_subset (optimum), 15

qcSubset, 16
replicates, 16
replicates(), 4

sd(), 9
sizes_batches (optimum), 15
spatial, 17
spatial(), 4

use_index, 18
use_index(), 3, 4, 9

19