# Package: splitTools (via r-universe)

September 8, 2024

**Title** Tools for Data Splitting

**Version** 1.0.1

**Description** Fast, lightweight toolkit for data splitting. Data sets can be partitioned into disjoint groups (e.g. into training, validation, and test) or into (repeated) k-folds for subsequent cross-validation. Besides basic splits, the package supports stratified, grouped as well as blocked splitting. Furthermore, cross-validation folds for time series data can be created. See e.g. Hastie et al. (2001) <doi:10.1007/978-0-387-84858-7> for the basic background on data partitioning and cross-validation.

**License** GPL (>= 2)

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**Imports** stats

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**URL** `https://mayer79.github.io/splitTools/`,
`https://github.com/mayer79/splitTools`

**BugReports** `https://github.com/mayer79/splitTools/issues`

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Repository** https://mayer79.r-universe.dev

**RemoteUrl** https://github.com/mayer79/splittools

**RemoteRef** HEAD

**RemoteSha** 5ec340fcb675a9e7cfb8f7f810e81aa8b7903f49

## Contents

---

`create_folds` *Create Folds*

---

### Description

This function provides a list of row indices used for k-fold cross-validation (basic, stratified, grouped, or blocked). Repeated fold creation is supported as well. By default, in-sample indices are returned.

### Usage

```
create_folds(
  y,
  k = 5L,
  type = c("stratified", "basic", "grouped", "blocked"),
  n_bins = 10L,
  m_rep = 1L,
  use_names = TRUE,
  invert = FALSE,
  shuffle = FALSE,
  seed = NULL
)
```

### Arguments

| | |
|---|---|
| `y` | Either the variable used for "stratification" or "grouped" splits. For other types of splits, any vector of the same length as the data intended to split. |
| `k` | Number of folds. |
| `type` | Split type. One of "stratified" (default), "basic", "grouped", "blocked". |
| `n_bins` | Approximate numbers of bins for numeric `y` (only for `type = "stratified"`). |
| `m_rep` | How many times should the data be split into k folds? Default is 1, i.e., no repetitions. |
| `use_names` | Should folds be named? Default is `TRUE`. |
| `invert` | Set to `TRUE` in order to receive out-of-sample indices. Default is `FALSE`, i.e., in-sample indices are returned. |
| `shuffle` | Should row indices be randomly shuffled within folds? Default is `FALSE`. |
| `seed` | Integer random seed. |

## Details

By default, the function uses stratified splitting. This will balance the folds regarding the distribution of the input vector `y`. (Numeric input is first binned into `n_bins` quantile groups.) If `type = "grouped"`, groups specified by `y` are kept together when splitting. This is relevant for clustered or panel data. In contrast to basic splitting, `type = "blocked"` does not sample indices at random, but rather keeps them in sequential groups.

## Value

If `invert = FALSE` (the default), a list with in-sample row indices. If `invert = TRUE`, a list with out-of-sample indices.

## See Also

`partition()`, `create_timefolds()`

## Examples

```
y <- rep(c(letters[1:4]), each = 5)
create_folds(y)
create_folds(y, k = 2)
create_folds(y, k = 2, m_rep = 2)
create_folds(y, k = 3, type = "blocked")
```

---

`create_timefolds` *Creates Folds for Time Series Data*

---

## Description

This function provides a list with in- and out-of-sample indices per fold used for time series k-fold cross-validation, see Details.

## Usage

```
create_timefolds(y, k = 5L, use_names = TRUE, type = c("extending", "moving"))
```

## Arguments

| | |
|---|---|
| `y` | Any vector of the same length as the data intended to split. |
| `k` | Number of folds. |
| `use_names` | Should folds be named? Default is `TRUE`. |
| `type` | Should in-sample data be "extending" over the folds (default) or consist of one single fold ("moving")? |

## Details

The data is first partitioned into $k + 1$ sequential blocks $B_1$ to $B_{k+1}$. Each fold consists of two index vectors: one with in-sample row numbers, the other with out-of-sample row numbers. The first fold uses $B_1$ as in-sample and $B_2$ as out-of-sample data. The second one uses either $B_2$ (if `type = "moving"`) or $\{B_1, B_2\}$ (if `type = "extending"`) as in-sample, and $B_3$ as out-of-sample data etc. Finally, the kth fold uses $\{B_1, ..., B_k\}$ ("extending") or $B_k$ ("moving") as in-sample data, and $B_{k+1}$ as out-of-sample data. This makes sure that out-of-sample data always follows in-sample data.

## Value

A nested list with in-sample and out-of-sample indices per fold.

## See Also

`partition()`, `create_folds()`

## Examples

```
y <- runif(100)
create_timefolds(y)
create_timefolds(y, use_names = FALSE)
create_timefolds(y, use_names = FALSE, type = "moving")
```

---

`multi_strata` *Create Strata from Multiple Features*

---

## Description

Creates a stratification vector based on multiple columns of a `data.frame` that can then be passed to the splitting functions.

Currently, the function offers two strategies to create the strata:

- "kmeans": k-means cluster analysis on scaled input. (Ordered factors are integer encoded first, unordered factors and character columns are one-hot-encoded.)
- "interaction": All combinations (after binning numeric columns into approximately `k` bins).

## Usage

```
multi_strata(df, strategy = c("kmeans", "interaction"), k = 3L)
```

## Arguments

| | |
|---|---|
| `df` | A `data.frame` used to form the stratification vector. |
| `strategy` | A string (either "kmeans" or "interaction") to compute the strata, see description. |
| `k` | An integer. For `strategy = "kmeans"`, it is the desired number of strata, while for `strategy = "interaction"`, it is the approximate number of bins per numeric feature before forming all combinations. |

## Value

Factor with strata as levels.

## See Also

`partition()`, `create_folds()`

## Examples

```
y_multi <- data.frame(
  A = rep(c(letters[1:4]), each = 20),
  B = factor(sample(c(0, 1), 80, replace = TRUE)),
  c = rnorm(80)
)
y <- multi_strata(y_multi, k = 3)
folds <- create_folds(y, k = 5)
```

---

`partition` *Split Data into Partitions*

---

## Description

This function provides row indices for data splitting, e.g., to split data into training, validation, and test. Different types of split strategies are supported, see Details. The partition indices are either returned as list with one element per partition (the default) or as vector of partition IDs.

## Usage

```
partition(
  y,
  p,
  type = c("stratified", "basic", "grouped", "blocked"),
  n_bins = 10L,
  split_into_list = TRUE,
  use_names = TRUE,
  shuffle = FALSE,
  seed = NULL
)
```

## Arguments

| | |
|---|---|
| `y` | Either the variable used for "stratification" or "grouped" splits. For other types of splits, any vector of the same length as the data intended to split. |
| `p` | A vector with split probabilities per partition, e.g., `c(train = 0.7, valid = 0.3)`. Names are passed to the output. |
| `type` | Split type. One of "stratified" (default), "basic", "grouped", "blocked". |
| `n_bins` | Approximate numbers of bins for numeric `y` (only for `type = "stratified"`). |

| | |
|---|---|
| `split_into_list` | Should the resulting partition vector be split into a list? Default is `TRUE`. |
| `use_names` | Should names of `p` be used as partition names? Default is `TRUE`. |
| `shuffle` | Should row indices be randomly shuffled within partition? Default is `FALSE`. Shuffling is only possible when `split_into_list = TRUE`. |
| `seed` | Integer random seed. |

## Details

By default, the function uses stratified splitting. This will balance the partitions as good as possible regarding the distribution of the input vector `y`. (Numeric input is first binned into `n_bins` quantile groups.) If `type = "grouped"`, groups specified by `y` are kept together when splitting. This is relevant for clustered or panel data. In contrast to basic splitting, `type = "blocked"` does not sample indices at random, but rather keeps them in groups: e.g., the first 80% of observations form a training set and the remaining 20% are used for testing.

## Value

A list with row indices per partition (if `split_into_list = TRUE`) or a vector of partition IDs.

## See Also

`create_folds()`

## Examples

```
y <- rep(c(letters[1:4]), each = 5)
partition(y, p = c(0.7, 0.3), seed = 1)
partition(y, p = c(0.7, 0.3), split_into_list = FALSE, seed = 1)
p <- c(train = 0.8, valid = 0.1, test = 0.1)
partition(y, p, seed = 1)
partition(y, p, split_into_list = FALSE, seed = 1)
partition(y, p, split_into_list = FALSE, use_names = FALSE, seed = 1)
partition(y, p = c(0.7, 0.3), type = "grouped")
partition(y, p = c(0.7, 0.3), type = "blocked")
```

# Index