

Package: impactR.utils (via r-universe)

October 9, 2026

Title Some utils for working with R
Version 0.0.0.9000
Description Providing functions that may be of common use.
Encoding UTF-8
Roxygen list(markdown = TRUE)
RoxygenNote 7.3.1
Imports dplyr (>= 1.1.0), rlang (>= 0.4.11), glue, stringr, purrr,
janitor, readr, tibble, rio, tidyr
Depends R (>= 2.10)
LazyData true
Suggests labelled, roxygen2
License MIT + file LICENSE
Config/pak/sysreqs make libicu-dev libssl-dev libx11-dev zlib1g-dev
Repository <https://impact-initiatives.r-universe.dev>
Date/Publication 2024-06-21 10:07:23 UTC
RemoteUrl <https://github.com/impact-initiatives/impactR.utils>
RemoteRef HEAD
RemoteSha 880a57ebb0282e377f888f352c746c9bde412e75

Contents

abort_bad_argument	2
count_missing_values	2
date_to_dm	3
deselect	3
df_col_start	4
df_diff	4
if_not_in_stop	5
import_csv	5
left_join_dup	6

left_joints	7
mutate_if_nulla	7
named_group_split	8
recode_values	8
rename_cols	9
row_optimum	9
row_string_count	10
subvec_in	11

Index	12
--------------	-----------

abort_bad_argument	<i>Abord bad argument</i>
--------------------	---------------------------

Description

Abord bad argument

Usage

abort_bad_argument(arg1, must, not = NULL, arg2 = NULL, same = NULL)

Arguments

- | | |
|------|---------------------------------|
| arg1 | An argument |
| must | What arg must be |
| not | Optional. What arg must not be. |
| arg2 | Another argument |
| same | The same type as arg1 |

Value

A stop statement

count_missing_values	<i>Count missing values for vars (by group)</i>
----------------------	---

Description

Count missing values for vars (by group)

Usage

count_missing_values(df, vars, group = NULL, pivot = TRUE)

Arguments

df	A data frame.
vars	Variable names.
group	Grouping column names. Default to NULL.
pivot	Logical. If TRUE, return a pivoted data frame.

date_to_dm	<i>Date to day and month</i>
------------	------------------------------

Description

Date to day and month

Usage

```
date_to_dm(vec)
```

Arguments

vec	A vector of type POSIXct.
-----	---------------------------

Value

The same vector with only the day and month in the env locale.

deselect	<i>Remove columns from a data frame</i>
----------	---

Description

Remove columns from a data frame

Usage

```
deselect(df, ...)
```

Arguments

df	A data frame.
...	Column names.

Value

A tibble with columns removed

df_col_start	<i>Get colnames that start, end, or both with a string</i>
--------------	--

Description

Get colnames that start, end, or both with a string

Usage

```
df_col_start(df, pattern_start)
```

```
df_col_end(df, pattern_end)
```

```
df_col_start_end(df, pattern_start, pattern_end)
```

Arguments

df	A data frame.
pattern_start	A start string
pattern_end	An end string.

Value

A vector of column names

df_diff	<i>Remove columns from a data frame that exists in another tibble</i>
---------	---

Description

Remove columns from a data frame that exists in another tibble

Usage

```
df_diff(df_a, df_b, ...)
```

Arguments

df_a	A data frame to remove columns from.
df_b	A data frame to extract columns names from.
...	Columns from df_b and df_a to keep in data frame.

Value

A tibble with some columns removed.

if_not_in_stop	<i>Stop statement "If not in colnames" with colnames</i>
----------------	--

Description

Stop statement "If not in colnames" with colnames

Usage

```
if_not_in_stop(df, cols, df_name, arg = NULL)
```

Arguments

df	A dataframe
cols	A vector of column names (quoted)
df_name	Provide the tibble name as a character string
arg	Default to NULL.

Value

A stop statement

import_csv	<i>Initial import with clean names (not an option!)</i>
------------	---

Description

Import a dataset from a CSV or a XLSX file. It ensures that the right type of column is guessed using the max value of guess_max by default.

Usage

```
import_csv(path, delim = ",", clean_names = TRUE, guess_max = 21474836)
```

```
import_xlsx(path, sheet = 1, clean_names = TRUE, guess_max = 21474836)
```

```
import_full_xlsx(path = NULL, clean_names = TRUE, guess_max = 21474836)
```

Arguments

path	The path to the file.
delim	Field separator, default is ",".
clean_names	Boolean. If TRUE, uses janitor::clean_names() to column clean names.
guess_max	Maximum guess of types. Default to 21474836.
sheet	A character string of the sheet name or the sheet position

Value

A dataframe.

Functions

- `import_csv()`: Import a .csv file.
- `import_xlsx()`: Import one sheet from a .xlsx.
- `import_full_xlsx()`: Import all sheets from a .xlsx.

left_join_dup	<i>Left join a list of dataframes</i>
---------------	---------------------------------------

Description

Left join a list of dataframes

Usage

```
left_join_dup(df_a, df_b, ..., duplicated_columns = 3)
```

Arguments

- | | |
|--------------------|--|
| df_a | A dataframe. |
| df_b | Another dataframe. |
| ... | Columns to join by. |
| duplicated_columns | Strategy between 1 and 4. Default to 3. See section "Treating duplicated columns". |

Value

A left-joined data frame.

Treating duplicated columns

There are 4 ways implemented on treating columns with the same names in both datasets:

- 1: an usual left-join with suffix being empty for df_a and ".dup_col" for df_b;
- 2: removing all duplicated columns from df_b, keeping only those of df_a;
- 3: remove the column from df_b if all values from df_b that are not NAs equal the values from df_a;
- 4: remove the column from df_b if all values from df_b are identical to values from df_a.

left_joints	<i>Left join a list of dataframes</i>
-------------	---------------------------------------

Description

`left_joints()` is a simple reducing functions based on `dplyr::left_join()`. `left_joints_dup()` is a reduce wrapper around `left_join_dup()`.

Usage

```
left_joints(list, ...)
```

```
left_joints_dup(list, ..., duplicated_columns = 3)
```

Arguments

`list` A list of data frames.

`...` Columns to join by.

`duplicated_columns`

Strategy between 1 and 4. Default to 3. See section "Treating duplicated columns".

Value

A left-joined data frame.

mutate_if_nulla	<i>Mutate with replacement if NULL or NA values</i>
-----------------	---

Description

Mutate with replacement if NULL or NA values

Usage

```
mutate_if_nulla(df, col, replacement)
```

Arguments

`df` A data frame.

`col` A column of df.

`replacement` A replacement of the same type as col.

Value

A mutated data frame.

named_group_split	<i>Named group split</i>
-------------------	--------------------------

Description

Named group split

Usage

```
named_group_split(df, group)
```

Arguments

df	A data frame.
group	Quoted column name to group by.

Value

A split and named list of data frames.

recode_values	<i>Replace values of a data frame's columns to one value</i>
---------------	--

Description

recode_values() recodes a singular value. recode_na() recodes usual values of NAs, for instance 999, NULL, NaN, and NA.

Usage

```
recode_values(df, values, to_value, ...)  
  
recode_na(df, ...)
```

Arguments

df	A data frame.
values	A vector of values.
to_value	A value.
...	Column names.

Details

If the column type is a character and the replacement a numeric, then the numeric is coerced to a character. If the column type is a numeric and the replacement is a character, then the column is coerced to character. NAs will remains NAs of the right type.

Value

A data frame with recoded values to one value.

rename_cols	<i>Rename columns</i>
-------------	-----------------------

Description

Rename columns

Usage

```
rename_cols(df, old_cols, new_cols)
```

Arguments

df	A data frame.
old_cols	A vector of old quoted variable names.
new_cols	A vector of new quoted variable names.

Value

Updated tibble with new names

row_optimum	<i>Row-wise optimum (default to pmax) of several columns</i>
-------------	--

Description

Row-wise optimum (default to pmax) of several columns

Usage

```
row_optimum(
  df,
  ...,
  optimum = "max",
  max_name = "pmax",
  min_name = "pmin",
  na_rm = TRUE,
  keep = "all"
)
```

Arguments

df	A data frame.
...	Numeric column names.
optimum	Should we calculate "max", "min" or "both"? Default to TRUE.
max_name	Column name for the mutated pmax.
min_name	Column name for the mutated pmin.
na_rm	Remove NAs. Default to TRUE.
keep	To be used within mutate. Default to "all".

Value

A data frame with pmax, pmin or both (and all columns or none, depending on "keep")

row_string_count	<i>Count the number of row-wise occurrences of a pattern</i>
------------------	--

Description

row_pattern_count() count row-wise occurrences of a pattern over the data frame; row_na_count() count row-wise number of NAs over the data frame.

Usage

```
row_string_count(df, pattern = "", new_colname = "count")

row_na_count(df, new_colname = "count")
```

Arguments

df	A data frame.
pattern	A pattern to pass to stringr::str_count(). Default to "dnk".
new_colname	The newly-mutated column name. Default to "count".

Value

A mutated data frame.

subvec_in	<i>Get a subset of a vector</i>
-----------	---------------------------------

Description

subvec_in() gets a subset of a vector if vector elements are in a given set. subvec_not_in() get a subset of a vector if vector elements are NOT in a given set.

Usage

```
subvec_in(vector, set)
```

```
subvec_not_in(vector, set)
```

Arguments

vector	A vector to subset.
--------	---------------------

set	A set-vector.
-----	---------------

Value

A subset of a list or a vector.

Index

`abort_bad_argument`, [2](#)

`count_missing_values`, [2](#)

`date_to_dm`, [3](#)

`deselect`, [3](#)

`df_col_end(df_col_start)`, [4](#)

`df_col_start`, [4](#)

`df_col_start_end(df_col_start)`, [4](#)

`df_diff`, [4](#)

`dplyr::left_join()`, [7](#)

`if_not_in_stop`, [5](#)

`import_csv`, [5](#)

`import_full_xlsx(import_csv)`, [5](#)

`import_xlsx(import_csv)`, [5](#)

`janitor::clean_names()`, [5](#)

`left_join_dup`, [6](#)

`left_join_dup()`, [7](#)

`left_joints`, [7](#)

`left_joints()`, [7](#)

`left_joints_dup(left_joints)`, [7](#)

`left_joints_dup()`, [7](#)

`mutate_if_nulla`, [7](#)

`named_group_split`, [8](#)

`recode_na(recode_values)`, [8](#)

`recode_values`, [8](#)

`rename_cols`, [9](#)

`row_na_count(row_string_count)`, [10](#)

`row_optimum`, [9](#)

`row_string_count`, [10](#)

`subvec_in`, [11](#)

`subvec_not_in(subvec_in)`, [11](#)