

Package: jrrosell (via r-universe)

October 15, 2026

Title Personal R package for Jordi Rosell

Version 0.0.0.9015

Description Useful functions for personal usage.

License CC0

URL <https://jrrosell.github.io/jrrosell>

BugReports <https://github.com/jrrosell/jrrosell/issues>

Depends R (>= 4.3.0)

Encoding UTF-8

Language en

Roxygen list(markdown = TRUE)

Suggests beeper, blastula, devtools, doFuture, doParallel, dplyr, ellmer, forcats, fst, future, ggplot2, glmnet, glue, hardhat, here, httr2, janitor, pak, parallel, pkgdown, rcmdcheck, readr, rstudioapi, sf, spacyr, stacks, stringr, styler, testthat, tibble, tictoc, udpipe, usethis, webfakes, workflowsets, yyjsonr

Imports curl, extrafont, jsonlite, openxlsx, parallelly, parsnip, purrr, recipes, rlang, scales, showtext, stringdist, stringi, sysfonts, tidyr, tm, tune, workflows

LazyData true

Config/roxygen2/version 8.0.0

Config/pak/sysreqs

cmake libfreetype6-dev make libicu-dev libpng-dev libuv1-dev libxml2-dev libssl-dev zlib1g-dev

Repository <https://jrrosell.r-universe.dev>

Date/Publication 2026-06-17 10:46:37 UTC

RemoteUrl <https://github.com/jrrosell/jrrosell>

RemoteRef HEAD

RemoteSha bc98e27b58c91e1892f726e36d1cf16f931fcff1

Contents

add_row_hash	3
addin_generate_documentation	3
as.bitstring	4
aside	5
calc_split_prop	5
calc_split_size	6
chars	7
check_installed_github	8
count_duplicated_rows	8
count_sorted	9
detect_cores	9
end_pipe	10
fit_results	10
fuzzy_token_set_ratio	11
generate_documentation	12
get_sentiments_by_language	12
glimpses	13
last_fit_metrics	14
name_unnamed_chunks	14
normalize_text	15
normalize_vec	15
notify_finished	16
package_github_name	17
package_github_version	17
plotBars	18
plot_histograms	18
plot_missing	19
plot_variable	19
prep_juice	20
prep_juice_ncol	20
prepare_docs	21
prepare_text	21
prepare_tokens	22
read_chr	23
read_url	24
read_xlsx	25
remove_stopwords	25
request_max_safe_cores_from_rss	26
resp_body_yyjson	27
sanitize_title_with_dashes	27
score_recipe	28
slugify	28
spain_ccaas	29
spain_provinces	29
sum_missing	30
summarize_n_distinct	30

Usage

```
addin_generate_documentation(
  model = "qwen2.5-coder:3b",
  context = rstudioapi::getActiveDocumentContext()
)
```

Arguments

`model` A single string with the ollama model to use.

`context` the IDE context. Defaults to `rstudioapi::getActiveDocumentContext`

Value

Nothing.

as.bitstring	<i>Data type utilities</i>
--------------	----------------------------

Description

Get the bit representation of a double number

Usage

```
as.bitstring(x)
```

Arguments

`x` A numeric vector.

Details

Get the bit representation of a double number Using `rev()` ensures that the bit order is correct, and the binary representation aligns with the usual convention of having the MSB first and the LSB last. This is because `numToBits()` returns the bits in the reverse order, and without `rev()`, we end up with the LSB first and the MSB last.

Source

<https://youtu.be/J4DnzjIFj8w>

Examples

```
0.1 + 0.2 == 0.3
as.bitstring(0.1 + 0.2)
as.bitstring(0.3)
```

`aside`*Multiple aside functions with base R pipe*

Description

Multiple aside functions with base R pipe

Usage

```
aside(x, ...)
```

Arguments

<code>x</code>	An object
<code>...</code>	functions to run aside

Examples

```
n_try <- 1
rnorm(200) |>
  matrix(ncol = 2) |>
  aside(
    print("Matrix prepared"),
    print(n_try)
  ) |>
  colSums()
```

`calc_split_prop`*Calculate split proportion*

Description

From a data frame, it returns the minimal split proportion for validation.

Usage

```
calc_split_prop(df, k = 10)
```

Arguments

<code>df</code>	A data frame
<code>k</code>	number of desired folds (default 10)

Details

The calc_validation_size function returns the optimal split proportion according to the number of rows for your validation set.

Source

<https://stats.stackexchange.com/a/305063/7387>

Examples

```
calc_split_prop(data.frame(row = 1:891))
```

calc_split_size	<i>Calculate split size</i>
-----------------	-----------------------------

Description

From binary classification problems, with the desired std_err it returns the minimal assesment/validation set size.

Usage

```
calc_split_size(  
  std_err = NULL,  
  confidence_interval = 0.95,  
  margin_error = 0.02  
)
```

Arguments

std_err	The desired std_err numeric (default NULL)
confidence_interval	(default 0.95)
margin_error	(default 0.02)

Details

The calc_validation_size function returns the minimal validation size for expected probabilities and desired error. s

Source

<https://stats.stackexchange.com/a/304996/7387>

Examples

```
calc_split_size()
calc_split_size(confidence_interval = 0.95, margin_error = 0.02)
calc_split_size(std_err = 0.02)
```

chars*Create a vector of characters from a string*

Description

Create a vector of characters from a string

Usage

```
chars(x, ...)
```

Arguments

x	a vector of characters of length 1.
...	unused

Details

chars expects a single string as input. To create a list of these, consider `lapply(strings, chars)`.

Value

a vector of characters

See Also

<https://github.com/jonocarroll/charcuterie>

Examples

```
chars("hola")
```

`check_installed_github`*Check if the last github version is installed*

Description

Check if the last main github version is installed.

Usage

```
check_installed_github(repo)
```

Arguments

repo a github repo/package. Ex: `check_installed_github("tidyverse/dplyr")`

Examples

```
if (FALSE) {  
  check_installed_github("jrosell/jrosell")  
}
```

`count_duplicated_rows` *Count the number of duplicated rows*

Description

Count the number of duplicated rows

Usage

```
count_duplicated_rows(df)
```

Arguments

df a data.frame

Examples

```
count_duplicated_rows(data.frame(a = c(1, 2, 3), b = c(3, 4, 5)))  
count_duplicated_rows(data.frame(a = c(1, 2, 3), b = c(1, 4, 5)))
```

count_sorted	<i>Count a variable or variables sorted</i>
--------------	---

Description

It returns the ordered counts of the variable in the data.frame.

Usage

```
count_sorted(df, ...)
```

Arguments

df	a data.frame
...	the variables to use and other arguments to count

Examples

```
data.frame(a = c("x", "y", "x"), b = c("z", "z", "n")) |>  
  count_sorted(a)
```

detect_cores	<i>Detect cores that could be used</i>
--------------	--

Description

Select cores in max/min of the available cores.

Usage

```
detect_cores(max = 10, min = 2)
```

Arguments

max	An integer with the max desired cores (default 10)
min	An integer with the min desired cores (default 2)

Details

The detect_cores function uses parallelly package. It returns the desired max cores if available or it fails if not min cores are available (excluding parallelly.availableCores.omit reserved cores or 1 if not defined).

Examples

```
cores <- detect_cores(max = 5, min = 1)  
print(cores)
```

end_pipe	<i>End a pipe</i>
----------	-------------------

Description

Print and expression and return invisible NULL at the end of a pipe.

Usage

```
end_pipe(x, expr)
```

Arguments

x	An object
expr	An expression

fit_results	<i>Fit a workflow with specific parameters</i>
-------------	--

Description

Fit a workflow with specific parameters

Usage

```
fit_results(wf, resamples, param_info = NULL, grid = 10, fn = "tune_grid", ...)
```

Arguments

wf	workflow
resamples	rset
param_info	for tune_* functions
grid	for tune_* functions
fn	the name of the function to run when tuning
...	Optional engine arguments

fuzzy_token_set_ratio *Fuzzy Token Set Ratio*

Description

This function computes a fuzzy similarity score between two strings based on the token set ratio methodology. It considers the intersection and differences between tokenized word sets from the input strings, and calculates a similarity score normalized by string lengths.

Usage

```
fuzzy_token_set_ratio(s1, s2, score_cutoff = 0)
```

Arguments

<code>s1</code>	A character string. The first string to compare.
<code>s2</code>	A character string. The second string to compare.
<code>score_cutoff</code>	A numeric value (default is 0) specifying the minimum similarity score threshold. Scores below this threshold may trigger early exits in the computation.

Details

This function performs the following steps:

- Tokenizes the input strings.
- Identifies intersecting and differing tokens between the two tokenized sets.
- Computes the longest common subsequence (LCS) distance for differing tokens and normalizes it.
- Calculates similarity ratios for intersecting tokens combined with differing token sets.
- Returns the maximum of the normalized LCS distance and the two intersecting token ratios.

The function short-circuits to return 100 if one token set is a subset of the other. If either input string is empty, the function returns 0.

Value

A numeric similarity score between 0 and 100, representing the degree of similarity between the two input strings.

Examples

```
# Example usage:  
fuzzy_token_set_ratio("fuzzy was a bear", "fuzzy was a dog", score_cutoff = 80)  
fuzzy_token_set_ratio("hello world", "world hello")
```

generate_documentation
<i>Generate documentation</i>

Description

It returns the generated documentation from the selected model

Usage

generate_documentation(x, model = "qwen2.5-coder:3b")

Arguments

- x A single string.
- model A single string with the ollama model to use.

Value

Generated documentation as a character string.

get_sentiments_by_language
<i>Get a sentiments by language</i>

Description

The multilingual sentiment lexicon was obtained from here on 2024-12-18 <https://aclanthology.org/P14-2063/>

Usage

get_sentiments_by_language(language = "en", lexicon = "chen_skiena")

Arguments

- language two letters language code.
- lexicon default and only valid value "chen_skiena"

Details

```
The files were generated this way: chen_skiena_lexicon <- bind_rows( here::here("P14-2063.Datasets",
"readable_neg_words_list.txt") |> read_delim(delim = " ", col_names = c("word", "lang")) |> mu-
tate(sentiment = factor("negative", levels = c("negative", "positive"))), here::here("P14-2063.Datasets",
"readable_pos_words_list.txt") |> read_delim(delim = " ", col_names = c("word", "lang")) |> mu-
tate(sentiment = factor("positive", levels = c("negative", "positive"))) ) chen_skiena_lexicon |>
write_fst( here::here("inst", "extdata", "chen_skiena_lexicon.fst"), compress = 100 ) top_languages
<- rlang::chr( ca = 'catalan', zh = 'chinese_simplified', da = 'danish', nl = 'dutch', en = 'english', eo
= 'esperanto', fi = 'finnish', fr = 'french', de = 'german', el = 'greek', hu = 'hungarian', it = 'italian',
la = 'latin', pt = 'portuguese', es = 'spanish', sv = 'swedish' ) nrc_lexicon <- read_delim("NRC-
Emotion-Lexicon-ForVariousLanguages.txt", delim = "\t") |> janitor::clean_names() |> pivot_longer(cols
= anger:trust, names_to = "sentiment") |> rename(english = english_word) |> pivot_longer(cols = -
c(sentiment, value), names_to = "language", values_to = "word") |> filter(sentiment %in% c("positive",
"negative")) |> filter(language %in% top_languages) |> transmute( sentiment = factor(sentiment,
c("negative", "positive")), language = factor(language, unique(language)), word = factor(word,
unique(word)), ) |> glimpse() nrc_lexicon |> write_fst( here::here("nrc_lexicon.fst"), compress =
100 )
```

Value

A tibble with word and sentiment columns

See Also

https://juliasilge.github.io/tidyttext/reference/get_sentiments.html

Examples

```
get_sentiments_by_language("ca")
```

glimpses

Glimpse multiple datasets

Description

Glimpse multiple datasets

Usage

```
glimpses(...)
```

Arguments

```
...           Multiple data.frame
```

Examples

```
df1 <- data.frame(a = c(1, 2))
df2 <- data.frame(b = c(3, 4))
glimpses(df1, df2)
```

last_fit_metrics	<i>Do the last fit and get the metrics</i>
------------------	--

Description

Do the last fit and get the metrics

Usage

```
last_fit_metrics(res, split, metric)
```

Arguments

res	Tune results
split	The initial split object
metric	What metric to use to select the best workflow

name_unnamed_chunks	<i>Name unnamed chunks in .Rmd or .qmd files</i>
---------------------	--

Description

Use with caution. It will overwrite your files.

Usage

```
name_unnamed_chunks(file_path)
```

Arguments

file_path	the file name
-----------	---------------

normalize_text	<i>Normalize text</i>
----------------	-----------------------

Description

This function processes a given text string by converting it to lowercase, removing numbers, non-alphanumeric characters, extra whitespace. It also transliterates text to ASCII, splits words, and reconstructs a clean text string suitable for analysis.

Usage

```
normalize_text(text, remove_digits = TRUE, remove_accents = TRUE)
```

Arguments

text	A character vector or object that can be coerced to a character string. Represents the input text to be cleaned.
remove_digits	= TRUE
remove_accents	= TRUE

Value

A normalized character vector

normalize_vec	<i>Center and scale double vectors</i>
---------------	--

Description

Center and scale double vectors

Usage

```
normalize_vec(...)
```

Arguments

...	a double vector or multiple double vectors
-----	--

Examples

```
normalize_vec(1, 2, 3, )
```

notify_finished	<i>Make a sound and send an email when a process finished</i>
-----------------	---

Description

The notify_finished make a sound using beep::beep, compose and email and send it returning the blastula::smtp_send call results.

Usage

```
notify_finished(name, body = "", ..., sound = 1, tictoc_result = NULL)
```

Arguments

name	The process name (Required)
body	The contents of the email (Default "")
...	Additional arguments to pass to the template function. If you're using the default template, you can use font_family to control the base font, and content_width to control the width of the main content; see blastula_template(). By default, the content_width is set to 1000px. Using widths less than 600px is generally not advised but, if necessary, be sure to test such HTML emails with a wide range of email clients before sending to the intended recipients. The Outlook mail client (Windows, Desktop) does not respect content_width.
sound	The sound for beep::beep call (Default 1)
tictoc_result	the result from tictoc::toc (Default NULL)

Details

The following environment variables should be set:

- MY_SMTP_USER from
- MY_SMTP_RECIPIENT to
- MY_SMTP_PASSWORD service password (for gmail you can use <https://myaccount.google.com/apppasswords>)
- MY_SMTP_PROVIDER blastula provider (gmail if not set)

Examples

```
if (exists("not_run")) {
  tictoc::tic()
  Sys.sleep(1)
  jrrosell::notify_finished("job", "Well done", sound = "fanfare", tictoc_result = tictoc::toc())
}
```

package_github_name	<i>Github name of the package</i>
---------------------	-----------------------------------

Description

Get the name of the package from the DESCRIPTION file of the master branch in the github repo

Usage

```
package_github_name(x, file_lines = NULL)
```

Arguments

x	a single repo/package to check Ex: package_github_name("tidyverse/dplyr")
file_lines	(default = NULL, internal)

Examples

```
if (FALSE) {  
  package_github_name("jrosell/jrrosell")  
}
```

package_github_version	<i>Github version of the package</i>
------------------------	--------------------------------------

Description

Get the version from the DESCRIPTION file of the master branch in the github repo

Usage

```
package_github_version(x, file_lines = NULL)
```

Arguments

x	a single repo/package to check Ex: package_github_version("tidyverse/dplyr")
file_lines	(default = NULL, internal)

Examples

```
if (FALSE) {  
  package_github_version("jrosell/jrrosell")  
}
```

plot_bars	<i>Plot bars for non double columns</i>
-----------	---

Description

Plot bars for non double columns

Usage

```
plot_bars(df, ..., top_values = 50)
```

Arguments

df	a data.frame
...	optional parameters to geom_histogram
top_values	first most common values (default 50)

Examples

```
plot_bars(data.frame(a = c("x", "y"), b = c("z", "z")))
```

plot_histograms	<i>Plot histograms for double columns</i>
-----------------	---

Description

Plot histograms for double columns

Usage

```
plot_histograms(df, ...)
```

Arguments

df	a data.frame
...	optional parameters to geom_histogram

Examples

```
plot_histograms(data.frame(a = c(1, 2), b = c(1, 3)))
```

plot_missing	<i>Plot missing values</i>
--------------	----------------------------

Description

Plot missing values

Usage

```
plot_missing(df)
```

Arguments

df	a data.frame
----	--------------

Examples

```
plot_missing(data.frame(a = c(1, NA), b = c(NA, 4)))
```

plot_variable	<i>Plot a variable</i>
---------------	------------------------

Description

It returns a bar or a histogram of the variable

Usage

```
plot_variable(df, variable, ..., type = "numeric")
```

Arguments

df	a data.frame
variable	the variable to use.
...	params passed to geom_*
type	numeric (default) or nominal.

Examples

```
data.frame(a = c("x", "y", "y"), b = c("z", "z", "x")) |> plot_variable(a)
```

prep_juice	<i>Prep, juice and glimpse a recipe or workflow</i>
------------	---

Description

Prep, juice and glimpse a recipe or workflow

Usage

```
prep_juice(object)
```

Arguments

object	A recipe or a workflow object with a recipe
--------	---

Source

<https://recipes.tidymodels.org/reference/update.step.html>

prep_juice_ncol	<i>Prep, juice and get cols from a recipe or workflow</i>
-----------------	---

Description

Prep, juice and get cols from a recipe or workflow

Usage

```
prep_juice_ncol(object)
```

Arguments

object	A recipe or a workflow object with a recipe
--------	---

prepare_docs	<i>Prepare docs for Analysis</i>
--------------	----------------------------------

Description

Prepare docs for Analysis

Usage

```
prepare_docs(df, ...)
```

Arguments

df	data frame with and id and text columns.
...	paramters passed to "prepare_tokens"

Value

A df with a list of tokens and character vector prepared_text columns for documents at column id and text at column "text"

Examples

```
# Example usage:
prepare_docs(data.frame(id = 1, text = "¡Hola! Esto es una prueba 123."))
```

prepare_text	<i>Prepare Text for Analysis</i>
--------------	----------------------------------

Description

This function processes a given text string by converting it to lowercase, removing numbers, non-alphanumeric characters, extra whitespace, and stopwords based on a specified language. It also transliterates text to ASCII, splits words, and reconstructs a clean text string suitable for analysis.

Usage

```
prepare_text(...)
```

Arguments

...	paramters passed to "prepare_tokens"
-----	--------------------------------------

Value

A cleaned character string, with stopwords removed and text formatted for analysis.

Examples

```
# Example usage:
prepare_text("¡Hola! Esto es una prueba 123.")
```

prepare_tokens	<i>Prepare tokens from text for Analysis</i>
----------------	--

Description

This function processes a given text string by converting it to lowercase, removing numbers, non-alphanumeric characters, extra whitespace, and stopwords based on a specified language. It also transliterates text to ASCII, splits words, and reconstructs a clean text string suitable for analysis.

Usage

```
prepare_tokens(
  text,
  stopwords = NULL,
  lang = "spanish",
  sep = "\\s+",
  remove_digits = TRUE,
  remove_accents = TRUE,
  lemmatize = c("none", "udpipe", "spacyr"),
  model_dir = getwd()
)
```

Arguments

text	A character vector or object that can be coerced to a character string. Represents the input text to be cleaned.
stopwords	A character vector specifying stopwords removal. Defaults tm:stopwords package.
lang	defaults to "spanish"
sep	separator for splitting defaults to "\\s+"
remove_digits	= TRUE
remove_accents	= TRUE
lemmatize	= c("none", "udpipe", "spacyr") defaults to "none"
model_dir	defaults to getwd()

Value

A cleaned character vector, with stopwords removed and text formatted for analysis and can be lemmatized optionally and then returns a character vector of lemmas.

read_chr	<i>Read character columns with clean names</i>
----------	--

Description

It's useful for reading the most common types of flat file data, comma separated values and tab separated values.

Usage

```
read_chr(
  file,
  delim = ",",
  locale = NULL,
  ...,
  date_names = "en",
  date_format = "%AD",
  time_format = "%AT",
  decimal_mark = ".",
  grouping_mark = "",
  tz = "CET",
  encoding = "UTF-8",
  asciify = FALSE
)
```

Arguments

file	Either a path to a file, a connection, or literal data (either a single string or a raw vector).
delim	Single character used to separate fields within a record.
locale	The locale controls defaults that vary from place to place. The default locale is US-centric (like R), but you can use <code>locale()</code> to create your own locale that controls things like the default time zone, encoding, decimal mark, big mark, and day/month names.
...	Other parameters to <code>readr::read_delim</code> .
date_names	"en" from <code>readr::locale</code>
date_format	"%AD" from <code>readr::locale</code>
time_format	"%AT" from <code>readr::locale</code>
decimal_mark	"." from <code>readr::locale</code>
grouping_mark	"" from <code>readr::locale</code>
tz	"CET"
encoding	"UTF-8"
asciify	FALSE

Details

The `read_chr` function works like `readr::read_delim`, except that column returned would be characters and with clean names. It requires `readr` and `janitor` packages installed.

Examples

```
read_chr(readr::readr_example("mtcars.csv"), delim = ",")
```

<code>read_url</code>	<i>Read the html text of an url</i>
-----------------------	-------------------------------------

Description

Read the HTML text of a URL with rate-limiting

Usage

```
read_url(url, sleep = 1, capacity = 1, realm = NULL)
```

Arguments

<code>url</code>	Full URL to request
<code>sleep</code>	Time (in seconds) to refill the bucket. Default: 1
<code>capacity</code>	Max requests per refill period. Default: 1 (i.e., one request every sleep seconds)
<code>realm</code>	Optional unique throttling scope. Defaults to domain of URL.

Details

It's useful for getting the text of webpages in a single character vector.

Value

HTML content as string or NULL on failure

Examples

```
if (FALSE) read_url("https://www.google.cat/", sleep = 1)
```

read_xlsx	<i>Read a sheet from a xlsx file into a tibble</i>
-----------	--

Description

It's useful for reading a single sheets from a Excel/Openoffice file.

Usage

```
read_xlsx(xlsxFile, ..., sheet = 1, startRow = 1)
```

Arguments

xlsxFile	The name of the file.
...	Other parameters to openxlsx::read.xlsx function
sheet	The name or index of the sheet (default 1)
startRow	The number of the starting reading row (default 1)

Details

The write_xlsx it's a wrapper for openxlsx::write.xlsx.

Examples

```
l <- list("IRIS" = iris, "MTCARS" = mtcars)
tmp_file <- tempfile(fileext = ".xlsx")
write_xlsx(l, tmp_file)
df <- read_xlsx(tmp_file)
file.remove(tmp_file)
```

remove_stopwords	<i>Remove stopwords</i>
------------------	-------------------------

Description

This function processes character vectors and remove the specified stop words or the stop words of the language from the tm package

Usage

```
remove_stopwords(text, stopwords = NULL, lang = "spanish")
```

Arguments

text	A character vector or object that can be coerced to a character string. Represents the input text to be cleaned.
stopwords	A character vector specifying stopwords removal. Defaults tm:stopwords package.
lang	defaults to "spanish"

Value

A character vector without stopwords

request_max_safe_cores_from_rss

Request the maximum safe number of cores

Description

When parallizing within resamples, required memory can crash the system.

Usage

```
request_max_safe_cores_from_rss(
  estimated_max_rss,
  memory_usage = 0.5,
  verbose = TRUE
)
```

Arguments

estimated_max_rss	butes of maximum rss it will used (You can get it from syrup package)
memory_usage	the proportion of the system memory that will be used (0.8)
verbose	to debug (TRUE)

Details

The detect_cores function uses parallely package. It returns the desired max cores if available or it fails if not min cores are available (excluding system reserved cores).

resp_body_yyjson	<i>Extract body from httr2 response using yyjsonr</i>
------------------	---

Description

Extract body from httr2 response using yyjsonr

Usage

```
resp_body_yyjson(resp, check_type = TRUE, simplifyVector = FALSE, ...)
```

Arguments

resp	A httr2::response object, created by httr2::req_perform().
check_type	Should the type actually be checked? Provided as a convenience for when using this function inside resp_body_* helpers.
simplifyVector	Should JSON arrays containing only primitives (i.e. booleans, numbers, and strings) be caused to atomic vectors?
...	Other parameters

sanitize_title_with_dashes	<i>Sanitize title with dashes</i>
----------------------------	-----------------------------------

Description

It generates slug URLs as WordPress does

Usage

```
sanitize_title_with_dashes(title)
```

Arguments

title	the title
-------	-----------

Examples

```
sanitize_title_with_dashes("Hello world")
```

score_recipe	<i>Tune a recipe using glmnet and lightgbm and stacks</i>
--------------	---

Description

Tune a recipe using glmnet and lightgbm and stacks

Usage

```
score_recipe(rec, resamples, grids = list(10, 10), metric = "accuracy")
```

Arguments

rec	recipe
resamples	rset
grids	for glmnet and lightgbm tuning
metric	to be compared

slugify	<i>Slugify character vectors</i>
---------	----------------------------------

Description

It generates slug URLs handling ASCII normalization

Usage

```
slugify(x)
```

Arguments

x	a character vector
---	--------------------

Examples

```
sanitize_title_with_dashes("Hello world")
```

spain_ccaas	<i>spain_ccaas</i>
-------------	--------------------

Description

spain_ccaas

Usage

spain_ccaas

Format

spain_ccaas:
A sf object with 19 rows and 4 columns:
OBJECTID
codigo
nombre
geometry

Source

https://github.com/koldLight/curso-r-dataviz/blob/master/dat/spain_ccaas.geojson

spain_provinces	<i>spain_provinces</i>
-----------------	------------------------

Description

spain_provinces

Usage

spain_provinces

Format

spain_provinces:
A sf object with 60 rows and 4 columns:
OBJECTID
codigo
nombre
geometry

Source

https://github.com/koldLight/curso-r-dataviz/blob/master/dat/spain_provinces.geojson

sum_missing	<i>Sum the missing values from a data.frame</i>
-------------	---

Description

Sum the missing values from a data.frame

Usage

```
sum_missing(...)
```

Arguments

... one or multiple data.frame

summarize_n_distinct	<i>Select constant columns from a data.frame</i>
----------------------	--

Description

Select constant columns from a data.frame

Usage

```
summarize_n_distinct(df)
```

Arguments

df a data.frame

tee	<i>Tee pipe that return the original value instead of the result</i>
-----	--

Description

Pipe a value forward into a function or call expression and return the original value instead of the result. This is useful when an expression is used for its side-effect, say plotting or printing.

Usage

```
tee(x, expr)
```

Arguments

x	An object
expr	An expression

Details

The tee pipe works like |>, except the return value is x itself, and not the result of expr call.

Thanks

I want to give credit to Michael Milton and Matthew Kay for the idea and the code.

Source

<https://mastodon.social/@multimeric@genomic.social/109555362766969210>

theme_roboto	<i>Sets a minimal theme using the Roboto font family</i>
--------------	--

Description

It requires roboto fonts installed in your O.S. and run z

Usage

```
theme_roboto(
  base_size = 13,
  strip_text_size = 14,
  strip_text_margin = 6,
  subtitle_size = 14,
  subtitle_margin = 10,
  plot_title_size = 18,
  plot_title_margin = 12,
  ...
)
```

Arguments

```
base_size      = 11
strip_text_size
               = 12
strip_text_margin
               = 5
subtitle_size  = 13
subtitle_margin
               = 10
```

plot_title_size	
	= 16
plot_title_margin	
	= 10
...	Other parameters passed to theme_set

theme_set_roboto_darkblue	
	<i>Sets a dark blue colored dark minimal theme using the Roboto font family</i>

Description

Sets a dark blue colored dark minimal theme using the Roboto font family

Usage

theme_set_roboto_darkblue(...)

Arguments

... Other parameters passed to theme_set

tokenize_text	<i>Tokenize text</i>
---------------	----------------------

Description

This function generates a character vector for a given text string

Usage

tokenize_text(text, sep = "\\s+")

Arguments

text	A character vector or object that can be coerced to a character string. Represents the input text to be cleaned.
sep	= "\\s+"

Value

A character vector

update_step	<i>Update recipe step values by id</i>
-------------	--

Description

Update the vaules of a specific recipe step located by id

Usage

```
update_step(object, target_id, ...)
```

Arguments

object	A recipe or a workflow object with a recipe
target_id	The id name of the step
...	The arguments to update the step.

workflow_boost_tree	<i>Create an xgboost tunable workflow for regression and classification</i>
---------------------	---

Description

Create an xgboost tunable workflow for regression and classification

Usage

```
workflow_boost_tree(rec, engine = "xgboost", counts = TRUE, ...)
```

Arguments

rec	prerocessing recipe to build the workflow
engine	xgboost, lightgbm (xgboost by default)
counts	Optional logic argument wether mtry use counts or not
...	optional engine arguments

workflow_elasticnet	Create a tuneable glmnet workflow for regression and classification
---------------------	---

Description

Create a tuneable glmnet workflow for regression and classification

Usage

```
workflow_elasticnet(rec, engine = "glmnet", ...)
```

Arguments

rec	prerocessing recipe to build the workflow
engine	glmnet, spark, brulee (glmnet by default)
...	Optional engine arguments

write_xlsx	Write a list of tibbles to a xlsx file
------------	--

Description

It's useful for saving multiple data to a multiple sheets of a single Excel/Openoffice/libreoffice file.

Usage

```
write_xlsx(data, distfile, ...)
```

Arguments

data	A named list of tibbles
distfile	The name of the destination file.
...	Other parameters to openxls::write.xlsx function

Details

The write_xlsx it's a wroapper for openxls::write.xlsx.

Index

- * **datasets**
 - spain_ccaas, 29
 - spain_provinces, 29
 - * **graphing**
 - theme_roboto, 31
 - theme_set_roboto_darkblue, 32
 - * **packages**
 - addin_generate_documentation, 3
 - check_installed_github, 8
 - generate_documentation, 12
 - package_github_name, 17
 - package_github_version, 17
 - * **parallel**
 - detect_cores, 9
 - request_max_safe_cores_from_rss, 26
 - * **pipe**
 - aside, 5
 - end_pipe, 10
 - tee, 30
 - * **processing**
 - name_unnamed_chunks, 14
 - notify_finished, 16
 - * **read**
 - read_chr, 23
 - read_url, 24
 - read_xlsx, 25
 - resp_body_yyjson, 27
 - * **text**
 - chars, 7
 - fuzzy_token_set_ratio, 11
 - get_sentiments_by_language, 12
 - normalize_text, 15
 - prepare_docs, 21
 - prepare_text, 21
 - prepare_tokens, 22
 - remove_stopwords, 25
 - tokenize_text, 32
 - * **tidymodels**
 - calc_split_prop, 5
 - calc_split_size, 6
 - fit_results, 10
 - last_fit_metrics, 14
 - normalize_vec, 15
 - prep_juice, 20
 - prep_juice_ncol, 20
 - score_recipe, 28
 - update_step, 33
 - workflow_boost_tree, 33
 - workflow_elasticnet, 34
 - * **types**
 - as.bitstring, 4
 - * **wrangling**
 - add_row_hash, 3
 - count_duplicated_rows, 8
 - count_sorted, 9
 - glimpses, 13
 - plotBars, 18
 - plot_histograms, 18
 - plot_missing, 19
 - plot_variable, 19
 - sanitize_title_with_dashes, 27
 - slugify, 28
 - sum_missing, 30
 - summarize_n_distinct, 30
 - * **write**
 - write_xlsx, 34
-
- add_row_hash, 3
 - addin_generate_documentation, 3
 - as.bitstring, 4
 - aside, 5
 - calc_split_prop, 5
 - calc_split_size, 6
 - chars, 7
 - check_installed_github, 8
 - count_duplicated_rows, 8
 - count_sorted, 9

detect_cores, [9](#)
end_pipe, [10](#)

fit_results, [10](#)
fuzzy_token_set_ratio, [11](#)

generate_documentation, [12](#)
get_sentiments_by_language, [12](#)
glimpses, [13](#)

last_fit_metrics, [14](#)

name_unnamed_chunks, [14](#)
normalize_text, [15](#)
normalize_vec, [15](#)
notify_finished, [16](#)

package_github_name, [17](#)
package_github_version, [17](#)
plot_bars, [18](#)
plot_histograms, [18](#)
plot_missing, [19](#)
plot_variable, [19](#)
prep_juice, [20](#)
prep_juice_ncol, [20](#)
prepare_docs, [21](#)
prepare_text, [21](#)
prepare_tokens, [22](#)

read_chr, [23](#)
read_url, [24](#)
read_xlsx, [25](#)
remove_stopwords, [25](#)
request_max_safe_cores_from_rss, [26](#)
resp_body_yyjson, [27](#)

sanitize_title_with_dashes, [27](#)
score_recipe, [28](#)
slugify, [28](#)
spain_ccaas, [29](#)
spain_provinces, [29](#)
sum_missing, [30](#)
summarize_n_distinct, [30](#)

tee, [30](#)
theme_roboto, [31](#)
theme_set_roboto_darkblue, [32](#)
tokenize_text, [32](#)

update_step, [33](#)

workflow_boost_tree, [33](#)
workflow_elasticnet, [34](#)
write_xlsx, [34](#)