Package: ThomasJeffersonUniv (via r-universe)

August 31, 2024

Type Package

Title Handy Tools for TJU/TJUH Employees

Version 0.1.3

Date 2024-07-01

Description Functions for admin needs of employees of Thomas Jefferson University and Thomas Jefferson University Hospital, Philadelphia, PA.

License GPL-2

Encoding UTF-8

Imports lubridate, stringi, stringdist, survival, timeDate, utils, writexl, zoo

Language en-US

Depends R (>= 4.4.0)

RoxygenNote 7.3.2

NeedsCompilation no

Author Tingting Zhan [aut, cre, cph] (<https://orcid.org/0000-0001-9971-4844>)

Maintainer Tingting Zhan <tingtingzhan@gmail.com>

Date/Publication 2024-07-01 17:50:09 UTC

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

RemoteUrl https://github.com/cran/ThomasJeffersonUniv

RemoteRef HEAD

RemoteSha d357b6ad95de57cd209988b6e912bfb425babb4c

Contents

`addProbs` *Conditional and/or Marginal Probabilities*

Description

Add conditional and/or marginal probabilities to a two-way contingency table.

Usage

```
addProbs(A, margin = seq_len(nd), fmt = "%d (%.1f%%)")
```

Arguments

`A`	matrix of typeof integer, two-dimensional contingency table. See addmargins
`margin`	integer scalar or vector, see addmargins
`fmt`	character scalar, C-style string format with a `%d` and an `%f%%` for the counts and proportions (order enforced).

Details

Function addProbs provides the joint, marginal (using `margin = 1:2`) and conditional (using `margin = 1L` or `margin = 2L`) probabilities of a two-dimensional contingency table.

Value

Function addProbs returns an 'addProbs' object, which inherits from table and noquote.

Note

margin.table (which is to be renamed as marginSums) is much slower than colSums.

The use of argument margin is the same as addmargins, and different from proportions!

See Also

rowSums colSums proportions

Examples

```
addProbs(table(warpbreaks$tension))

storage.mode(VADeaths) = 'integer'
addProbs(VADeaths)
addProbs(VADeaths, margin = 1L)
rowSums(proportions(VADeaths, margin = 1L))
addmargins(VADeaths, margin = 1L)
```

---

anniversary *Number of Anniversaries Between Two Dates*

Description

Number of anniversaries between two dates.

Usage

```
anniversary(to, from)
```

Arguments

to	an R object convertible to POSIXlt, end date/time
from	an R object convertible to POSIXlt, start date/time

Details

1. Year difference between from and to dates are calculated
2. In either situation below, subtract one (1) year from the year difference obtained in Step 1.
   - Month of from is later than month of to;
   - Months of from and to are the same, but day of from is later than day of to.

   In either of such situations, the anniversary of the current year has not been reached.
3. If any element from Step 2 is negative, stop.

Value

Function anniversary returns an integer scalar or vector.

`asDifftime` *Create Time Differences, Extended*

Description

To create difftime object with additional time units `'months'` and `'years'`.

Usage

```
asDifftime(
  tim,
  units = names(timeUnits()),
  negative_do = stop(sQuote(deparse1(substitute(tim))), " has negative value!"),
  ...
)
```

Arguments

`tim`	numeric or difftime object, similar usage as in function as.difftime
`units`	character scalar, similar usage as in function as.difftime, but with additional options `'months'` and `'years'`
`negative_do`	exception handling if input `tim` has negative element(s). Default is to stop
`...`	additional parameters, currently not in use

Details

Function asDifftime improves function as.difftime in terms that

- If input `tim` is a difftime object, function units_difftime<- is called and the unit of `tim` is updated. In function as.difftime, `tim` is returned directly, i.e., parameter `units` is ignored
- Time units `'months'` and `'years'` are supported, in addition to `'secs'`, `'mins'`, `'hours'`, `'days'`, `'weeks'` supported in function as.difftime. Moreover, partial matching (via function match.arg) is allowed, while function as.difftime requires exact matching.
- End user may choose to stop if `tim` has negative values. Function as.difftime does not check for negative `tim`.

Value

Function asDifftime returns a difftime object.

Note

Potential name clash with function as_difftime

bibentry2rmd *R Markdown Format of citation and/or bibentry*

Description

R markdown format of a citation and/or bibentry object.

Usage

```
bibentry2rmd(x = "R")
```

Arguments

x	character scalar, 'R' (default) or name of an R package

Details

Function bibentry2rmd beautifies the output from function utils:::format.bibentry (with option style = 'text') in the following ways.

- Line break '\n' is replaced by a white space;
- Fancy quotes “, ”, ‘ and ’ are removed;
- doi entries are shown as URLs with labels (in R markdown grammar).

Value

Function bibentry2rmd returns a character scalar or vector.

Examples

```
bibentry2rmd('survival')
if (FALSE) { # disabled for ?devtools::check
ap = rownames(installed.packages())
lapply(ap, FUN = bibentry2rmd)
}
```

checkCount *Positive Counts in a logical vector*

Description

Number and percentage of positive counts in a logical vector.

Usage

```
checkCount(x)
```

Arguments

x	logical vector

Value

Function checkCount returns a character scalar.

Examples

```
checkCount(as.logical(infert$case))
```

---

checkDuplicated *Inspect Duplicated Records in a data.frame*

Description

To inspect duplicated records in a data.frame.

Usage

```
checkDuplicated(
  data,
  f,
  dontshow = character(length = 0L),
  file = tempfile(pattern = "checkDuplicated_", fileext = ".xlsx"),
  ...
)
```

Arguments

data	data.frame
f	formula, criteria of duplication, e.g., use ~ mrn to identify duplicated mrn, or use ~ mrn + visitdt to identify duplicated mrn:visitdt
dontshow	(optional) character scalar or vector, variable names to be omitted in output diagnosis file
file	character scalar, path of diagnosis file, print out of substantial duplicates
...	additional parameters, currently not in use

Value

Function checkDuplicated returns a data.frame.

Examples

```
(d1 = data.frame(A = c(1, 1), B = c(NA_character_, 'text')))


(d2 = data.frame(A = c(1, 2), B = c(NA_character_, 'text')))
```

`date_difftime_` *Concatenate a Date and a difftime Object*

Description

..

Usage

```
date_difftime_(date_, difftime_, tz = "UTC", tol = sqrt(.Machine$double.eps))
```

Arguments

`date_`	an R object containing Date information
`difftime_`	a difftime object
`tz`	character scalar, time zone, see as.POSIXlt.Date and ISOdatetime
`tol`	numeric scalar, tolerance in finding second. Default `sqrt(.Machine$double.eps)` as in all.equal.numeric

Value

Function date_difftime_ returns a POSIXct object.

Note

For now, I do not know how to force function `readxl::read_excel` to read a column as POSIXt. By default, such column will be read as difftime.

See `lubridate:::date.default` for the handling of year and month!

Examples

```
(x = as.Date(c('2022-09-10', '2023-01-01', NA, '2022-12-31')))
y = as.difftime(c(47580.3, NA, 48060, 30660), units = 'secs')
units(y) = 'hours'
y
date_difftime_(x, y)
```

date_time_ *Concatenate Date and Time*

Description

Concatenate date and time information from two objects.

Usage

```
date_time_(date_, time_)
```

Arguments

date_	an R object containing Date information
time_	an R object containing time (POSIXt) information

Details

Function date_time_ is useful as clinicians may put date and time in different columns.

Value

Function date_time_ returns a POSIXct object.

Examples

```
(today = Sys.Date())
(y = ISOdatetime(year = c(1899, 2010), month = c(12, 3), day = c(31, 22),
  hour = c(15, 3), min = 2, sec = 1, tz = 'UTC'))
date_time_(today, y)
```

hexavigesimalExcel *Hexavigesimal (Base 26L) and Excel Columns*

Description

Convert between decimal, hexavigesimal in C-style, and hexavigesimal in Excel-style.

Usage

```
Excel2int(x)

Excel2C(x)
```

Arguments

x	character scalar or vector, which consists of (except missingness) only letters A to Z and a to z.

Details

Convert between decimal, hexavigesimal in C-style, and hexavigesimal in Excel-style.

Decimal	0	1	25	26	27	51	52	676	702	703
Hexavigesimal; C	0	1	P	10	11	1P	20	100	110	111
Hexavigesimal; Excel	0	A	Y	Z	AA	AY	AZ	YZ	ZZ	AAA

Function Excel2C converts from hexavigesimal in Excel-style to hexavigesimal in C-style.

Function Excel2int converts from hexavigesimal in Excel-style to decimal, using function Excel2C and strtoi.

Value

Function Excel2int returns an integer vector.

Function Excel2C returns a character vector.

References

http://mathworld.wolfram.com/Hexavigesimal.html

See Also

as.hexmode

Examples

```
int1 = c(NA_integer_, 1L, 25L, 26L, 27L, 51L, 52L, 676L, 702L, 703L)
Excel1 = c(NA_character_, 'A', 'Y', 'Z', 'AA', 'AY', 'AZ', 'YZ', 'ZZ', 'AAA')
C1 = c(NA_character_, '1', 'P', '10', '11', '1P', '20', '100', '110', '111')
stopifnot(identical(int1, Excel2int(Excel1)), identical(int1, strtoi(C1, base = 26L)))

int2 = c(NA_integer_, 1L, 4L, 19L, 37L, 104L, 678L)
Excel2 = c(NA_character_, 'a', 'D', 's', 'aK', 'cZ', 'Zb')
stopifnot(identical(int2, Excel2int(Excel2)))
Excel2C(Excel2)

head(swiss[Excel2int('A')])
```

matchDF *Match Rows of One data.frame to Another*

Description

To match the rows of one data.frame to the rows of another data.frame.

Usage

```
matchDF(
  x,
  table = unique.data.frame(x),
  by = names(x),
  by.x = character(),
  by.table = character(),
  view.table = character(),
  trace = FALSE,
  ...
)
```

Arguments

x	data.frame, the rows of which to be matched.
table	data.frame, the rows of which to be matched *against*.
by	character scalar or vector
by.x, by.table	character scalar or vector
view.table	(optional) character scalar or vector, variable names of table to be printed in fuzzy suggestion (if applicable)
trace	logical scalar, to provide detailed diagnosis information, default FALSE
...	additional parameters, currently not in use

Value

Function matchDF returns a integer vector

Note

Unfortunately, R does not provide case-insensitive match. Only case-insensitive grep methods are available.

Examples

```
DF = swiss[sample(nrow(swiss), size = 55, replace = TRUE), ]
matchDF(DF)
```

mergeDF *An Alternative Merge Operation*

Description

..

Usage

```
mergeDF(
  x,
  table,
  by = character(),
  by.x = character(),
  by.table = character(),
  ...
)
```

Arguments

x	data.frame, on which new columns will be added. All rows of x will be retained in the returned object, *in their original order*.
table	data.frame, columns of which will be added to x. Not all rows of table will be included in the returned object
by	character scalar or vector
by.x, by.table	character scalar or vector
...	additional parameters of matchDF

Value

Function mergeDF returns a data.frame.

Note

We avoid merge.data.frame as much as possible, because it's slow and even sort = FALSE may not completely retain the original order of input x.

Examples

```
# examples inspired by ?merge.data.frame

(authors = data.frame(
 surname = c('Tukey', 'Venables', 'Tierney', 'Ripley', 'McNeil'),
 nationality = c('US', 'Australia', 'US', 'UK', 'Australia'),
 deceased = c('yes', rep('no', 4))))
(books = data.frame(
 name = c('Tukey', 'Venables', 'Tierney', 'Ripley',
```

```
  'Ripley', 'McNeil', 'R Core', 'Diggle'),
 title = c(
  'Exploratory Data Analysis',
  'Modern Applied Statistics',
  'LISP-STAT', 'Spatial Statistics', 'Stochastic Simulation',
  'Interactive Data Analysis', 'An Introduction to R',
  'Analysis of Longitudinal Data'),
 other.author = c(
  NA, 'Ripley', NA, NA, NA, NA, 'Venables & Smith',
  'Heagerty & Liang & Scott Zeger')))

(m = mergeDF(books, authors, by.x = 'name', by.table = 'surname'))
attr(m, 'nomatch')
```

---

phone10 *10-digit US phone number*

Description

..

Usage

```
phone10(x, sep = "")
```

Arguments

x	character vector
sep	character scalar

Details

Function phone10 converts all US and Canada (+1) phone numbers to 10-digit.

Value

Function phone10 returns a character vector of nchar-10.

Examples

```
x = c(
 '+1(800)275-2273', # Apple
 '1-888-280-4331', # Amazon
 '000-000-0000'
)
phone10(x)
phone10(x, sep = '-')
```

rbinds *Row-Bind a list of data.frame*

Description

..

Usage

```
rbinds(x, make.row.names = FALSE, ..., .id = "idx")
```

Arguments

x	a list of named data.frame
make.row.names, ...	additional parameters of rbind.data.frame
.id	character value to specify the name of ID column, nomenclature follows rbindlist

Details

Yet to look into ggplot2:::rbind_dfs closely.

Mine is slightly slower than the fastest alternatives, but I have more checks which are useful.

Value

Function rbinds returns a data.frame.

References

https://stackoverflow.com/questions/2851327/combine-a-list-of-data-frames-into-one-data-frame

Examples

```
x = list(A = swiss[1:3, 1:2], B = swiss[5:9, 1:2]) # list of 'data.frame'
rbinds(x)
rbinds(x, make.row.names = TRUE)
```

`sample.by.int` *Indices of Stratified Sampling*

Description

Indices of Stratified Sampling

Usage

```
sample.by.int(f, ...)
```

Arguments

`f`	factor
`...`	potential parameters of sample.int

Details

End user should use interaction to combine multiple factors.

Value

Function sample.by.int returns an integer vector.

See Also

`dplyr::slice_sample`

Examples

```
id1 = sample.by.int(state.region, size = 2L)
state.region[id1]

id2 = sample.by.int(f = with(npk, interaction(N, P)), size = 2L)
npk[id2, c('N', 'P')] # each combination selected 2x
```

`sign2` *Sign of Difference of Two Objects*

Description

..

Usage

```
sign2(
  e1,
  e2,
  name1 = substitute(e1),
  name2 = substitute(e2),
  na.detail = TRUE,
  ...
)
```

Arguments

`e1`, `e2`	two R objects, must be both numeric vectors, or ordered factors with the same levels
`name1`, `name2`	two language objects, or character scalars
`na.detail`	logical scalar, whether to provide the missingness details of `e1` and `e2`. Default `TRUE`.
`...`	additional parameters, currently not in use

Details

Function sign2 extends sign in the following ways

- two ordered factors can be compared;
- (detailed) information on missingness are provided.

Value

Function sign2 returns character vector when `na.detail = TRUE`, or ordered factor when `na.detail = FALSE`.

Examples

```
lv = letters[c(1,3,2)]
x0 = letters[1:3]
x = ordered(sample(x0, size = 100, replace = TRUE), levels = lv)
y = ordered(sample(x0, size = 50, replace = TRUE), levels = lv)
x < y # base R ok
pmax(x, y) # base R okay
```

```
pmin(x, y) # base R okay
x[c(1,3)] = NA
y[c(3,5)] = NA
table(sign(unclass(y) - unclass(x)))
table(sign2(x, y))
table(sign2(x, y, na.detail = FALSE), useNA = 'always')
```

`sourcePath` *Source All R Files under a Directory*

Description

source all `*.R` and `*.r` files under a directory.

Usage

```
sourcePath(path, ...)
```

Arguments

`path`	character scalar, parent directory of `.R` files
`...`	additional parameters of source

Value

Function sourcePath does not have a returned value

`splitDF` *Split data.frame by Row*

Description

split.data.frame into individual rows.

Usage

```
splitDF(x)
```

Arguments

`x`	data.frame

Value

Function splitDF returns a list of nrow-1 data.frames.

Note

We use split.data.frame with argument f being attr(x, which = 'row.names', exact = TRUE) instead of seq_len(.row_names_info(x, type = 2L)), not only because the former is faster, but also .rowNamesDF<- enforces that row.names.data.frame must be unique.

Examples

```
splitDF(head(mtcars)) # data.frame with rownames
splitDF(head(warpbreaks)) # data.frame without rownames
splitDF(data.frame()) # exception
```

---

subset_ *Inspect a Subset of data.frame*

Description

..

Usage

```
subset_(x, subset, select, select_pattern, avoid, avoid_pattern)
```

Arguments

x	a data.frame
subset	logical expression, see function subset.data.frame
select	character vector, columns to be selected, see function subset.data.frame
select_pattern	regular expression regex for multiple columns to be selected
avoid	character vector, columns to be avoided
avoid_pattern	regular expression regex, for multiple columns to be avoided

Details

Function subset_ is different from subset.data.frame, such that

- if both select and select_pattern are missing, only variables mentioned in subset are selected;
- be able to select all variables, except those in avoid and avoid_pattern;
- always returns data.frame, i.e., forces drop = FALSE

Value

Function subset_ returns a data.frame, with additional attributes

attr(,'vline') integer scalar, position of a vertical line (see ?flextable::vline)

attr(,'jhighlight)' character vector, names of columns to be flextable::highlighted.

Examples

```
subset_(trees, Girth > 9 & Height < 70)
subset_(swiss, Fertility > 80, avoid = 'Catholic')
subset_(warpbreaks, wool == 'K')
```

`Surv_3Date` *Create Surv Object using Three Dates*

Description

Create right-censored Surv object using start, stop and censoring dates.

Usage

```
Surv_3Date(start, stop, censor, units = "years", ...)
```

Arguments

`start`, `stop`, `censor`	Date, POSIXlt or POSIXct object
`units`	(optional) character scalar, time units
`...`	potential parameters, currently not in use

Value

Function Surv_3Date returns a Surv object.

Examples

```
library(survival)
d1 = within(survival::udca, expr = {
  edp_yr = Surv_3Date(entry.dt, death.dt, last.dt, units = 'years')
  edp_mon = Surv_3Date(entry.dt, death.dt, last.dt, units = 'months')
})
head(d1)

noout = within(survival::udca, expr = {
  edp_bug = Surv_3Date(entry.dt, death.dt, as.Date('1991-01-01'), units = 'months')
})
subset(survival::udca, subset = entry.dt > as.Date('1991-01-01')) # check error as suggested
```

TJU_Cayuse *Award & Effort from Cayuse*

Description

Print out grant and effort from Cayuse.

Usage

```
aggregateAwards(path = "~/Downloads", fiscal.year = year(Sys.Date()))

viewProposal(path = "~/Downloads", fiscal.year = year(Sys.Date()))

viewAward(path = "~/Downloads")

award2LaTeX(path = "~/Downloads")
```

Arguments

path	character scalar, directory of downloaded award .csv file. Default is the download directory '~/Downloads'
fiscal.year	integer scalar

Details

- go to https://jefferson.cayuse424.com/sp/index.cfm
- My Proposals -> Submitted Proposals. Lower-right corner of screen, 'Export to CSV'. Downloaded file has name pattern '^proposals_.*\\.csv'
- My Awards -> Awards (*not* 'Active Projects'). Lower-right corner of screen, 'View All', then 'Export to CSV'. Downloaded file has name pattern '^Awards_.*\\.csv'
- My Awards -> Awards. Click into each project, under 'People' tab to find my 'Sponsored Effort'

Function aggregateAwards aggregates grant over different period (e.g. from Axx-xx-001, Axx-xx-002, Axx-xx-003 to Axx-xx). Then we need to manually added in our 'Sponsored Effort' in the returned .csv file.

Value

..

Examples

```
if (FALSE) {
aggregateAwards()
viewAward()
viewProposal()
```

```
award2LaTeX()
}
```

`TJU_Fiscal_Year` *TJU Fiscal Year*

Description

..

Usage

```
TJU_Fiscal_Year(x)
```

Arguments

x	integer scalar

Value

Function TJU_Fiscal_Year returns a length-two Date vector, indicating the start (July 1 of the previous calendar year) and end date (June 30) of a fiscal year.

Examples

```
TJU_Fiscal_Year(2022L)
```

`TJU_SchoolTerm` *TJU School Term*

Description

..

Usage

```
TJU_SchoolTerm(x)
```

Arguments

x	Date object

Value

TJU_SchoolTerm returns a character vector

Examples

```
TJU_SchoolTerm(as.Date(c('2021-03-14', '2022-01-01', '2022-05-01')))
```

`TJU_Workday` *Thomas Jefferson University Workdays*

Description

To summarize the number of workdays, weekends, holidays and vacations in a given time-span (e.g., a month or a quarter of a year).

Usage

```
TJU_Workday(x, vacations)
```

Arguments

`x`	character scalar or vector (e.g., `'2021-01'` for January 2021, `'2021 Q1'` for 2021 Q1 (January to March)), or integer scalar or vector (e.g., `2021L` for year 2021); The time-span to be summarized. Objects of classes yearqtr and yearmon are also accepted.
`vacations`	Date vector, vacation days

Details

Function TJU_Workday summarizes the workdays, weekends, Jefferson paid holidays (New Year's Day, Martin Luther King, Jr. Day, Memorial Day, Fourth of July, Labor Day, Thanksgiving and Christmas) and your vacation (e.g., sick, personal, etc.) days (if any), in a given time-span.

Per Jefferson policy (source needed), if a holiday is on Saturday, then the preceding Friday is considered to be a weekend day. If a holiday is on Sunday, then the following Monday is considered to be a weekend day.

Value

Function TJU_Workday returns a factor.

Examples

```
table(TJU_Workday(c('2021-01', '2021-02')))

tryCatch(TJU_Workday(c('2019-10', '2019-12')), error = identity)
table(c(TJU_Workday('2019-10'), TJU_Workday('2019-12'))) # work-around

table(TJU_Workday('2022-12'))

table(TJU_Workday('2022 Q1', vacations = seq.Date(
 from = as.Date('2022-03-14'), to = as.Date('2022-03-18'), by = 1)))
```

```
table(TJU_Workday('2022 Q2', vacations = as.Date(c(
 '2022-05-22', '2022-05-30', '2022-06-01', '2022-07-04'))))

table(TJU_Workday(2021L))
```

`trimws_` *Remove Leading/Trailing and Duplicated (Symbols that Look Like) White Spaces*

Description

To remove leading/trailing and duplicated (symbols that look like) white spaces.

More aggressive than function trimws.

Usage

```
trimws_(x)
```

Arguments

`x`	an object with typeof being character

Details

Function trimws_ is more aggressive than trimws, that it removes

- duplicated white spaces
- symbols that look like white space, such as `\u00a0` (no-break space)

Value

Function trimws_ returns an object of typeof character.

Note

gsub keeps attributes

Examples

```
(x = c(A = ' a  b  ', b = 'a .  s', ' a  ,  b ; ', '\u00a0  ab '))
base::trimws(x)
# raster::trim(x) # do not want to 'Suggests'
trimws_(x)

(xm = matrix(x, nrow = 2L))
trimws_(xm)
```

```
#library(microbenchmark)
#microbenchmark(trimws(x), trimws_(x))
```

---

zip5 *5-digit US Zip Code*

Description

..

Usage

```
zip5(x)
```

Arguments

x	character vector

Details

Function zip5 converts all US zip codes to 5-digit.

Value

Function zip5 returns a character vector of nchar-5.

Examples

```
zip5(c('14901', '41452-1423'))
```

Index