Package 'janitor'

January 4, 2018

Title Simple Tools for Examining and Cleaning Dirty Data

Version 0.3.1

Description The main janitor functions can: perfectly format data.frame column names; provide quick one- and two-variable tabulations (i.e., frequency tables and crosstabs); and isolate duplicate records. Other janitor functions nicely format the tabulation results. These tabulate-and-report functions approximate popular features of SPSS and Microsoft Excel. This package follows the principles of the ``tidyverse" and works well with the pipe function %>%. janitor was built with beginning-to-intermediate R users in mind and is optimized for user-friendliness. Advanced R users can already do everything covered here, but with janitor they can do it faster and save their thinking for the fun stuff.

URL https://github.com/sfirke/janitor

BugReports https://github.com/sfirke/janitor/issues

Depends R (>= 3.1.2)

Imports dplyr (>= 0.5.0), tidyr, magrittr

License MIT + file LICENSE

LazyData true

RoxygenNote 6.0.1

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Sam Firke [aut, cre],
Chris Haid [ctb],
Ryan Knight [ctb]

Maintainer Sam Firke <samuel.firke@gmail.com>

Repository CRAN

Date/Publication 2018-01-04 21:49:02 UTC

R topics documented:

`add_totals_col` *Append a totals column to a data.frame.*

Description

This function is deprecated, use `adorn_totals` instead.

Usage

```
add_totals_col(dat, na.rm = TRUE)
```

Arguments

`dat`	an input data.frame with at least one numeric column.
`na.rm`	should missing values (including NaN) be omitted from the calculations?

Value

Returns a data.frame with a totals column containing row-wise sums.

add_totals_row *Append a totals row to a data.frame.*

Description

This function is deprecated, use `adorn_totals` instead.

Usage

```
add_totals_row(dat, fill = "-", na.rm = TRUE)
```

Arguments

`dat`	an input data.frame with at least one numeric column.
`fill`	if there are more than one non-numeric columns, what string should fill the bottom row of those columns?
`na.rm`	should missing values (including NaN) be omitted from the calculations?

Value

Returns a data.frame with a totals row, consisting of "Total" in the first column and column sums in the others.

adorn_crosstab *Add presentation formatting to a crosstabulation table.*

Description

Formats a data.frame containing counts of co-occurrences of two variables (i.e., a contingency table or crosstab). Adds a mix of percentages, Ns, totals row/column, and custom rounding to a table of integer counts, in the style of a Microsoft Excel PivotTable. The result is no longer clean data, but is an audience-friendly way to report results.

Designed to run on the output of a call to `janitor::crosstab`, but can be called on any data.frame containing a contingency table, e.g., the result of `dplyr::count()` followed by `tidyr::spread()`.

Usage

```
adorn_crosstab(dat, denom = "row", show_n = TRUE, digits = 1,
  show_totals = FALSE, rounding = "half to even")
```

Arguments

dat	a data.frame with row names in the first column and numeric values in all other columns. Usually the piped-in result of a call to `crosstab` that included the argument `percent = "none"`.
denom	the denominator to use for calculating percentages. One of "row", "col", or "all".
show_n	should counts be displayed alongside the percentages?
digits	how many digits should be displayed after the decimal point?
show_totals	display a totals summary? Will be a row, column, or both depending on the value of `denom`.
rounding	method to use for truncating percentages - either "half to even", the base R default method, or "half up", where 14.5 rounds up to 15.

Value

Returns a data.frame.

Examples

```
mtcars %>%
 crosstab(gear, cyl) %>%
 adorn_crosstab(denom = "all")

# showing with all parameters
mtcars %>%
 crosstab(gear, cyl) %>%
 adorn_crosstab(., denom = "col", rounding = "half up", show_n = FALSE, digits = 2)
mtcars %>%
 crosstab(cyl, am) %>%
 adorn_crosstab(., denom = "all", digits = 0, rounding = "half up")
```

---

adorn_totals *Append a totals row and/or column to a data.frame.*

Description

This function excludes the first column of the input data.frame, assuming it's a descriptive variable not to be summed. It also excludes other non-numeric columns.

Usage

```
adorn_totals(dat, which = c("row", "col"), fill = "-", na.rm = TRUE)
```

Arguments

dat	an input data.frame with at least one numeric column.
which	one of "row", "col", or c("row", "col")
fill	if there are multiple non-numeric columns, what string should fill the bottom row of those columns?
na.rm	should missing values (including NaN) be omitted from the calculations?

Value

Returns a data.frame augmented with a totals row, column, or both.

Examples

```
mtcars %>%
  crosstab(am, cyl) %>%
  adorn_totals()
```

---

clean_names *Cleans names of a data.frame.*

Description

Resulting names are unique and consist only of the _ character, lowercase letters, and numbers.

Usage

```
clean_names(dat)
```

Arguments

dat	the input data.frame.

Value

Returns the data.frame with clean names.

Examples

```
# not run:
# clean_names(poorly_named_df)

# library(readxl)
# not run:
# readxl("messy_excel_file.xlsx") %>% clean_names()
```

convert_to_NA *Convert string values to true* NA *values.*

Description

Converts instances of user-specified strings into NA. Can operate on either a single vector or an entire data.frame.

Usage

```
convert_to_NA(dat, strings)
```

Arguments

dat	vector or data.frame to operate on.
strings	character vector of strings to convert.

Value

Returns a cleaned object. Can be a vector, data.frame, or `tibble::tbl_df` depending on the provided input.

Warning

Deprecated, do not use in new code. Use `dplyr::na_if()` instead.

See Also

janitor_deprecated

crosstab *Generate a crosstabulation of two vectors.*

Description

Create a crosstab, displaying either frequencies or percentages calculated by row, column, or overall.

`crosstab` can be called in two ways:

1) It can simply be called on two vectors, like `crosstab(mtcars$gear, mtcars$cyl)`.

2) Or, when both vectors are columns in a single data.frame, the data.frame can be provided as the first argument, followed by two unquoted column names to crosstabulate. This enables passing in a data.frame from a `%>%` pipeline, in addition to making for a shorter function call. Like `mtcars %>% crosstab(gear, cyl)`.

For fancy formatting of the resulting data.frame, see `adorn_crosstab`.

Usage

```
crosstab(...)

## Default S3 method:
crosstab(vec1, vec2, percent = "none", show_na = TRUE,
  ...)

## S3 method for class 'data.frame'
crosstab(.data, ...)
```

Arguments

`...`	additional arguments, if calling `crosstab` on a data.frame.
`vec1`	the vector to place on the crosstab column. If supplying a data.frame, this should be an unquoted column name.
`vec2`	the vector to place on the crosstab row. If supplying a data.frame, this should be an unquoted column name.
`percent`	which grouping to use for percentages, if desired (defaults to "none", which returns simple counts). Must be one of "none", "row", "col", or "all".
`show_na`	a logical value indicating whether counts should be displayed where either variable is `NA`.
`.data`	(optional) a data.frame, in which case `vec1` and `vec2` should be unquoted column names.

Value

Returns a data.frame with the frequencies of the crosstabulated variables.

Examples

```
# Calling on two vectors:
a <- c("hi", "hi", "lo", "lo")
b <- c(1, 2, 2, 2)
crosstab(a, b)

crosstab(mtcars$cyl, mtcars$gear)
crosstab(mtcars$cyl, mtcars$gear, "row")

# Passing in a data.frame using a pipeline:
mtcars %>% crosstab(cyl, gear)
mtcars %>% crosstab(cyl, gear, "row")

# This allows for upstream operations
# prior to the crosstab() call:
library(dplyr)
mtcars %>%
  filter(am == 0) %>%
  crosstab(cyl, gear)
```

excel_numeric_to_date *Convert dates encoded as serial numbers to Date class.*

Description

Converts numbers like `42370` into date values like `2016-01-01`.

Defaults to the modern Excel date encoding system. However, Excel for Mac 2008 and earlier Mac versions of Excel used a different date system. To determine what platform to specify: if the date 2016-01-01 is represented by the number 42370 in your spreadsheet, it's the modern system. If it's 40908, it's the old Mac system. More on date encoding systems at http://support.office.com/en-us/article/Date-calculations-in-Excel-e7fe7167-48a9-4b96-bb53-5612a800b487.

Usage

```
excel_numeric_to_date(date_num, date_system = "modern")
```

Arguments

`date_num`	numeric vector of serial numbers to convert.
`date_system`	the date system, either `"modern"` or `"mac pre-2011"`.

Value

Returns a vector of class Date.

Examples

```
excel_numeric_to_date(40000)
```

get_dupes *Get rows of a* `data.frame` *with identical values for the specified variables.*

Description

For hunting duplicate records during data cleaning. Specify the data.frame and the variable combination to search for duplicates and get back the duplicated rows.

Usage

```
get_dupes(dat, ...)
```

Arguments

`dat`	the input data.frame.
`...`	unquoted variable names to search for duplicates.

Value

Returns a data.frame (actually a `tbl_df`) with the full records where the specified variables have duplicated values, as well as a variable `dupe_count` showing the number of rows sharing that combination of duplicated values.

Examples

```
get_dupes(mtcars, mpg, hp)
# or called with the magrittr pipe %>% :
mtcars %>% get_dupes(wt)
```

---

janitor_deprecated *Deprecated Functions in Package janitor*

Description

These functions are provided for compatibility with older versions of `janitor` only, and may be defunct as soon as the next release.

Details

- `use_first_valid_of`
- `convert_to_NA`

---

ns_to_percents *Convert a numeric data.frame to row-, column-, or totals-wise percentages.*

Description

This function excludes the first column of the input data.frame, assuming that it contains a descriptive variable.

Usage

```
ns_to_percents(dat, denom = "row", na.rm = TRUE, total_n = NULL)
```

Arguments

`dat`	a data.frame with row names in the first column and numeric values in all other columns.
`denom`	the denominator to use for calculating percentages. One of "row", "col", or "all".
`na.rm`	should missing values (including NaN) be omitted from the calculations?
`total_n`	an optional number to use as the denominator when calculating table-level percentages (when denom = "all"). Supply this if your input data.frame `dat` has values that would throw off the denominator if they were included, e.g., if there's a totals row appended to the bottom of the table.

Value

Returns a data.frame of percentages, expressed as numeric values between 0 and 1.

Examples

```
mtcars %>%
  crosstab(am, cyl) %>%
  ns_to_percents(denom = "all")

  # when total_n is needed
  mtcars %>%
  crosstab(am, cyl) %>%
  adorn_totals("row") %>% # add a totals row that should not be included in the denominator
  ns_to_percents(denom = "all", total_n = nrow(mtcars)) # specify correct denominator
```

---

remove_empty_cols *Removes empty columns from a data.frame.*

Description

Removes all columns from a data.frame that are composed entirely of NA values.

Usage

```
remove_empty_cols(dat)
```

Arguments

dat	the input data.frame.

Value

Returns the data.frame with no empty columns.

Examples

```
# not run:
# dat %>% remove_empty_cols
```

remove_empty_rows *Removes empty rows from a data.frame.*

Description

Removes all rows from a data.frame that are composed entirely of NA values.

Usage

```
remove_empty_rows(dat)
```

Arguments

dat	the input data.frame.

Value

Returns the data.frame with no empty rows.

Examples

```
# not run:
# dat %>% remove_empty_rows
```

tabyl *Generate a frequency table from a vector.*

Description

Create a frequency table of a variable, returned as a data.frame. It shows counts, percentages and, if NA values are present, valid percentages (calculated excluding NA values). A fully-featured alternative to `table()`.

`tabyl` can be called in two ways:

1) It can simply be called on a vector, like `tabyl(mtcars$gear)`.

2) A data.frame can be provided as the first argument, followed by an unquoted column name to tabulate. This enables passing in a data.frame from a %>% pipeline, like `mtcars %>% tabyl(gear)`.

Usage

```
tabyl(...)

## Default S3 method:
tabyl(vec, sort = FALSE, show_na = TRUE, ...)

## S3 method for class 'data.frame'
tabyl(.data, ...)
```

Arguments

...	additional arguments, if calling tabyl on a data.frame.
vec	the vector to tabulate. If supplying a data.frame, this should be an unquoted column name.
sort	a logical value indicating whether the resulting table should be sorted in descending order of n.
show_na	a logical value indicating whether the count of NA values should be displayed, along with an additional column showing valid percentages.
.data	(optional) a data.frame, in which case vec should be an unquoted column name.

Value

Returns a data.frame with the frequencies and percentages of the tabulated variable.

Examples

```
# Calling on a vector:
val <- c("hi", "med", "med", "lo")
tabyl(val)
tabyl(mtcars$cyl, sort = TRUE)

# Passing in a data.frame using a pipeline:
mtcars %>% tabyl(cyl, sort = TRUE)

# illustrating show_na functionality:
my_cars <- rbind(mtcars, rep(NA, 11))
tabyl(my_cars$cyl)
tabyl(my_cars$cyl, show_na = FALSE)
```

---

top_levels *Generate a frequency table of a factor grouped into top-n, bottom-n, and all other levels.*

Description

Get a frequency table of a factor variable, grouped into categories by level.

Usage

```
top_levels(input_vec, n = 2, show_na = FALSE, sort = FALSE)
```

Arguments

input_vec	the factor variable to tabulate.
n	number of levels to include in top and bottom groups
show_na	should cases where the variable is NA be shown?
sort	should the resulting table be sorted in descending order?

Value

Returns a data.frame (actually a `tbl_df`) with the frequencies of the grouped, tabulated variable. Includes counts and percentages, and valid percentages (calculated omitting `NA` values, if present in the vector and `show_na = TRUE`.)

Examples

```
top_levels(as.factor(mtcars$hp), 2)
```

---

use_first_valid_of *Returns first non-NA value from a set of vectors.*

Description

At each position of the input vectors, iterates through in order and returns the first non-NA value. This is a robust replacement of the common `ifelse(!is.na(x), x, ifelse(!is.na(y), y, z))`. It's more readable and handles problems like `ifelse`'s inability to work with dates in this way.

Usage

```
use_first_valid_of(..., if_all_NA = NA)
```

Arguments

`...`	the input vectors. Order matters: these are searched and prioritized in the order they are supplied.
`if_all_NA`	what value should be used when all of the vectors return `NA` for a certain index? Default is NA.

Value

Returns a single vector with the selected values.

Warning

Deprecated, do not use in new code. Use `dplyr::coalesce()` instead.

See Also

janitor_deprecated

Index