Package 'tibbleColumns'

June 5, 2018

Title A useful set of functions that focus on tibble data frames.

Version 0.1

Description Designed to offer some time saving functions that fix problems you didn't even know you had within the tidyverse. It also introduces some advanced methods I've developed for advanced pipe sequences.

Depends R (>= 3.4.1)

License GPL

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1.9000

Suggests knitr, rmarkdown

VignetteBuilder knitr

R topics documented:

bool_to_binary *New binary numeric column mirroring logical column*

Description

Look at a logical TRUE FALSE column and make new column with binary representation.

Usage

```
bool_to_binary(df, col, remove_bool_col = FALSE)
```

Arguments

df, col	a data frame and logical TRUE FALSE column to change to binary

Examples

```
iris %>% mutate(Setosa = str_detect(.$Species, "setosa")) %>% bool_to_binary(.,Setosa)
```

change_XoX_column *General X over X change Column*

Description

Creates a change column based on integer or numeric column

Usage

```
change_XoX_column(df, col1, col2, XoX)
```

Arguments

df, col1, col2, XoX	a data frame two columnnames and XoX name for new column

Examples

```
change_XoX_column(mtcars, drat, wt, "MoM")
```

change_XoX_column_group *General X over X Change Column by Group*

Description

Creates a change column based on a group. This function is specific as the data must have three columns at most. A category group column group a/b, device type, segment, a calendar group month, year, day and a numeric column to aggregate users, visits, clicks etc.. The data columns MUST be in that order as well. Category Group, Calendar Group, Numeric Aggregate.

Usage

```
change_XoX_column_group(df, col1, col2, XoX)
```

Arguments

df, col1, col2, XoX	a data frame two columnnames and XoX name for new column

Examples

```
tb %>% select(Type, Month, Users) %>% change_XoX_column_group(Dec,Jan,"MoM")
```

file_choose *Tidy wrapper for file.choose function with doc format options*

Description

Opens GUI to select specific file and read in as either csv or xls with a sheet option for xls file.

Usage

```
file_choose(type, sheet = NULL)
```

Arguments

type	format type for read in document either csv or xls

Examples

```
file_choose("csv")
```

lag_col *Create lag column based on another*

Description

Select a column to have the lag row of be noted in a new mutated column.

Usage

```
lag_col(df, col, replace_na_with = NULL)
```

Arguments

`df, col, replace_na_with`
: a data frame a column and what to replace NAs with

Examples

```
mtcars %>% lead_col(cyl,0)
```

lead_col *Create lead column based on another*

Description

Select a column to have the lead row of be noted in a new mutated column.

Usage

```
lead_col(df, col, replace_na_with = NULL)
```

Arguments

`df, col, replace_na_with`
: a data frame a column and what to replace NAs with

Examples

```
mtcars %>% lead_col(cyl,0)
```

lm_summary_tibble *Linear Model Summary Tibble*

Description

This function returns a tidy tibble output of the most important parts of a lm summary a la the broom package.

Usage

```
lm_summary_tibble(df, dep)
```

Arguments

df, dep	a dataframe and a dependent variable name

Examples

```
mtcars %>% select(mpg,cyl,wt) %>% lm_summary_tibble(mpg)
```

prop_column *General Proportion Column*

Description

This function creates a proportion column based on a column specified.

Usage

```
prop_column(df, col)
```

Arguments

df, col	a data frame and a column name

Examples

```
mtcars %>% count(cyl, disp) %>% arrange(desc(n)) %>% prop_column(n)
```

prop_column_group *Proportion Column by Group*

Description

Groups one column, adds a column for count of each group and adds a column for proportion of total based on count

Usage

```
prop_column_group(df, group)
```

Arguments

df, group	a dataframe and a group column name

Examples

```
mtcars %>% prop_column_group(cyl)
```

replace_all_na *Replace all na's in tibble with a value*

Description

Replace NA's with a zero for dataframes. Defaults with a 0 unless something else typed.

Usage

```
replace_all_na(df, replace_with = NULL)
```

Arguments

df, replace_with	a data frame and what to replace with

Examples

```
mtcars %>% tbl_out("cars") %>% sjmisc::set_na(na = 0) %>% replace_all_na()
```

tbl_lookup *Get a list of groups in certain column*

Description

Designate a column and get a single column listing the groups in that column.

Usage

```
tbl_lookup(df, ...)
```

Arguments

df, ...	a data frame and a column or columns to get group list

Examples

```
mtcars %>% tbl_out("cars") %>% tbl_module(filter(.,hp > 150),"fastCars") %>% tbl_lookup(cyl) %>% tbl_out("cylL
```

tbl_module *Run a function outside of the pipe sequence*

Description

Run a function outside of the pipe sequence and create a tibble in global environment for it while passing previous state of data frame through to the next pipe step.

Usage

```
tbl_module(df, fun, name)
```

Arguments

df, fun, name	a data frame, a function to run and a name for created tibble object

Examples

```
mtcars %>% tbl_out("cars") %>% tbl_module(filter(.,hp > 150),"fastCars") %>% tbl_lookup(cyl) %>% tbl_out("cylL
```

tbl_out — *Shorter alias for tibble_out function - Tibble a data frame state within a pipe series*

Description

Create a tibble for the state of a data frame within a pipe series and assign it as an object to the global environment.

Usage

```
tbl_out(df, name, suppress = FALSE)
```

Arguments

df, name	a data frame and a name for created tibble object

Examples

```
mtcars %>% group_by(cyl) %>% prop_column_group(cyl) %>% tbl_out("grouped") %>% filter(Count >9)
```

tibble_out — *Tibble a data frame state within a pipe series*

Description

Create a tibble for the state of a data frame within a pipe series and assign it as an object to the global environment.

Usage

```
tibble_out(df, name, suppress = FALSE)
```

Arguments

df, name	a data frame and a name for created tibble object
suppress	prevents the function from creating the tibble in the parent environment

Examples

```
mtcars %>% group_by(cyl) %>% prop_column_group(cyl) %>% tibble_out("grouped") %>% filter(Count >9)
```

`ttest_tibble` *A tidy t.test summary tibble*

Description

Allows to pass a tibble data_frame to the base R t.test function over two numeric columns. Then extracts the output statistics and outputs a tibble.

Usage

```
ttest_tibble(df1, df2)
```

Arguments

`df1, df2`	two tibble dataframes

Examples

```
ttest_tibble(t1$num,t2$num)
```

Index