Package ‘iNZightTools’

February 23, 2020

Type Package

Title Tools for ‘iNZight’

Version 1.8.3

Imports utils, stats, methods, survey, grDevices, readr (>= 1.2.0),
readxl, haven, magrittr, tibble, dplyr,forcats, tidyr,
stringr, tools, chron, lubridate, zoo, validate

Suggests testthat, covr, RCurl

BugReports https://github.com/iNZightVIT/iNZightTools/issues

Contact inzight_support@stat.auckland.ac.nz

URL https://www.stat.auckland.ac.nz/~wild/iNZight/

Description Provides a collection of wrapper functions for common variable and dataset manipulation workflows primarily used by 'iNZight', a graphical user interface providing easy exploration and visualisation of data for students of statistics, available in both desktop and online versions. Additionally, many of the functions return the 'tidyverse' code used to obtain the result in an effort to bridge the gap between GUI and coding.

License GPL-3

Encoding UTF-8

Language en-GB

RoxygenNote 7.0.2

NeedsCompilation no

Author Tom Elliott [aut, cre] (<https://orcid.org/0000-0002-7815-6318>),
Christoph Knopf [ctb],
Akshay Gupta [ctb],
Owen Jin [ctb] (Tidyverse variable/data manipulation functions),
Lushi Cai [ctb],
Yiwen He [ctb] (Dates/times and dataset manipulation),
Daniel Barnett [ctb] (Data validation)

Maintainer Tom Elliott <tom.elliott@auckland.ac.nz>

Repository CRAN

Date/Publication 2020-02-23 11:30:03 UTC
R topics documented:

- add_suffix ............................................. 3
- aggregateData .......................................... 4
- aggregatedt ........................................... 5
- appendrows ........................................... 5
- code ...................................................... 6
- collapseLevels ......................................... 7
- combineCatVars ......................................... 8
- convertToCat .......................................... 8
- convert_to_datetime .................................... 9
- countMissing ........................................... 10
- createNewVar .......................................... 10
- create_varname ......................................... 11
- deleteVars ............................................ 12
- extract_part ........................................... 12
- filterLevels ........................................... 13
- filterNumeric .......................................... 14
- filterRandom ........................................... 15
- filterRows ............................................. 16
- fitDesign ............................................... 16
- fitModel ................................................. 17
- iNZighTools ............................................ 18
- is_cat .................................................... 18
- is_dt ..................................................... 19
- is_num .................................................... 19
- is_preview ............................................. 20
- is_survey ............................................... 20
- is_svydesign .......................................... 21
- is_svyrep ............................................. 21
- joindata ............................................... 22
- load_rda ............................................... 22
- missingToCat .......................................... 23
- newdevice ............................................. 24
- rankVars ............................................... 24
- read_meta .............................................. 25
- read_text .............................................. 26
- renameLevels ......................................... 26
- renameVars ............................................ 27
- reorderLevels ......................................... 28
- reshape_data .......................................... 29
- save_rda ............................................... 30
- separate ................................................ 30
- sheets ................................................... 31
- smart_read ............................................. 32
- sortVars ............................................... 32
- stackVars .............................................. 33
- standardizeVars ....................................... 34
add_suffix

Description

When creating new variables or modifying the data set, we often add a suffix added to distinguish the new name from the original one. However, if the same action is performed twice (for example, filtering a data set), the suffix is duplicated (data.filtered.filtered). This function averts this by adding the suffix if it doesn’t exist, and otherwise appending a counter (data.filtered2).

Usage

add_suffix(name, suffix)

Arguments

name a character vector containing (original) names
suffix the suffix to add, a length-one character vector

Value

character vector of names with suffix appended

Examples

add_suffix("data", "filtered")
add_suffix(c("data.filtered", "data.filtered.reshaped"), "filtered")
**aggregateData**

*Aggregate data by categorical variables*

**Description**

Aggregate a dataframe into summaries of all numeric variables by grouping them by specified categorical variables and returns the result along with tidyverse code used to generate it.

**Usage**

`aggregateData(.data, vars, summaries)`

**Arguments**

- `.data` a dataframe to aggregate
- `vars` a character vector of categorical variables in `.data` to group by
- `summaries` summaries to generate for the groups generated in `vars`. Valid summaries are "iqr", "mean", "median", "sd", "sum"

**Value**

aggregated dataframe containing the summaries with tidyverse code attached

**Author(s)**

Owen Jin

**See Also**

`code`

`countMissing`

**Examples**

```r
aggregated <- aggregateData(iris, vars = c("Species"), summaries = c("mean", "sd", "iqr"))
cat(code(aggregated))
head(aggregated)
```
aggregatedt

Aggregate datetimes

Description
Aggregate datetimes

Usage
aggregatedt(.data, method, key, name)

Arguments
- .data: dataframe or tibble to aggregate
- method: the type of aggregation
- key: the key column
- name: the name of the variable

Value
a data frame/tibble

Author(s)
Yiwen He

appendrows

Append row to the dataset

Description
Append row to the dataset

Usage
appendrows(.data, imported_data, date)

Arguments
- .data: original dataset
- imported_data: imported dataset
- date: whether a "When_Added" column is required
Value

dataset with new rows appended

Author(s)

Yiwen He

Description

Used to grab code from a data.frame generated by this package.

Usage

code(data)

Arguments

data dataset you want to extract the code from

Details

This is simply a helper function to grab the contents of the ‘code’ attribute contained in the data object.

Value

The code used to generate the data.frame, if available (else NULL)

Author(s)

Tom Elliott
collapseLevels

Description

Collapse several values in a categorical variable into one level

Usage

collapseLevels(
  .data,
  var,
  levels,
  collapse = paste(levels, collapse = " "),
  name = sprintf("%s.coll", var)
)

Arguments

.data a dataframe to collapse
var a character of the name of the categorical variable to collapse
levels a character vector of the levels to be collapsed
collapse name of the newly created level
name a name for the new variable

Value

the original dataframe containing a new column of the collapsed variable with tidyverse code attached

Author(s)

Owen Jin

See Also

code

Examples

collapsed <- collapseLevels(iris, var = "Species",
  levels = c("setosa", "virginica"))
head(collapsed)
combineCatVars  Combine categorical variables into one

Description
Combine specified categorical variables by concatenating their values into one character, and returns the result along with tidyverse code used to generate it.

Usage
combineCatVars(.data, vars, sep = ".", name = paste(vars, collapse = sep))

Arguments
- .data: a dataframe with the columns to be combined
- vars: a character vector of the categorical variables to be combined
- sep: the separator to combine the values of the variables in var by. "." by default
- name: a name for the new variable

Value
original dataframe containing a new column of the renamed categorical variable with tidyverse code attached

Author(s)
Owen Jin

Examples
combined <- combineCatVars(warpbreaks, vars = c("wool", "tension"), sep = ".")
cat(code(combined))
head(combined)

convertToCat  Convert numeric variables to categorical

Description
Convert specified numeric variables into factors

Usage
convertToCat(.data, vars, names = paste(vars, "cat", sep = "."))
**convert_to_datetime**

### Description
Convert to datetime

### Usage
```r
convert_to_datetime(.data, factorname, convname, newname)
```

### Arguments
- `.data`: dataframe
- `factorname`: name of the variable
- `convname`: format
- `newname`: name of the new column

### Value
- dataframe with datetime column

---

**convert_to_datetime**  
*Convert to datetime*

### Description
Convert to datetime

### Usage
```r
convert_to_datetime(.data, factorname, convname, newname)
```

### Arguments
- `.data`: dataframe
- `factorname`: name of the variable
- `convname`: format
- `newname`: name of the new column

### Value
- dataframe with datetime column
countMissing  

*Description*

Count missing values

*Usage*

```r
countMissing(var, na.rm = FALSE)
```

*Arguments*

- `var`  
  the vector to sum up the number of missing values
- `na.rm`  
  ignore this

*Value*

the number of missing values for that vector

*Author(s)*

Owen Jin

*See Also*

`aggregateData`

createNewVar  

*Description*

Create a new variable by using a valid R expression and returns the result along with tidyverse code used to generate it.

*Usage*

```r
createNewVar(.data, new_var = "new.variable", R_exp)
```
create_varname

Arguments

.data a dataframe to which to add a new variable to
.new_var a character of the new variable name. "new.variable" by default
.R_exp a character of a valid R expression which can generate a vector of values

Value

original dataframe containing the new column created from R_exp with tidyverse code attached

Author(s)

Owen Jin

See Also

code

Examples

cat(code(created))
head(created)
Examples

```r
create_varname("a new variable")
create_varname("8d4-2q5")
```

deleteVars  

Delete variables  

Description
Delete variables from a dataset

Usage  
deleteVars(.data, vars)

Arguments

- `.data` dataset  
- `vars` variables to delete

Value

dataset without chosen variables

Author(s)

Tom Elliott

extract_part  

Extract part of a datetimes variable  

Description
Extract part of a datetimes variable

Usage  
extract_part(.data, varname, part, name)

Arguments

- `.data` dataframe  
- `varname` name of the variable  
- `part` part of the variable wanted  
- `name` name of the new column
filterLevels

Value

dataframe with extracted part column

Author(s)

Yiwen He

filterLevels  Filter data by levels of a categorical variables

Description

Filter a dataframe by some levels of one categorical variable and returns the result along with
tidyverse code used to generate it.

Usage

filterLevels(.data, var, levels)

Arguments

.data  a dataframe to filter
var  character of the column in .data to filter by
levels  a character vector of levels in var to filter by

Value

filtered dataframe with tidyverse code attached

Author(s)

Owen Jin

See Also

code

Examples

filtered <- filterLevels(iris, var = "Species",
levels = c("versicolor", "virginica"))
cat(code(filtered))
head(filtered)
filterNumeric

Filter data by levels of a numeric variable

Description

Filter a dataframe by some boolean condition of one numeric variable and returns the result along with tidyverse code used to generate it.

Usage

```r
filterNumeric(.data, var, op, num)
```

Arguments

- `.data`: a dataframe to filter
- `var`: character of the column in `.data` to filter by
- `op`: a logical operator of "="","",">=",";">"","==" or "!=" for the boolean condition
- `num`: a number for which the `op` applies to

Value

filtered dataframe with tidyverse code attached

Author(s)

Owen Jin

See Also

code

Examples

```r
filtered <- filterNumeric(iris, var = "Sepal.Length", op = "<=" , num = 5)  
cat(code(filtered))
head(filtered)
```
Description

Take a specified number of groups of observations with fixed group size by sampling without replacement and returns the result along with tidyverse code used to generate it.

Usage

```
filterRandom(.data, n, sample_size)
```

Arguments

- `.data`: a dataframe to sample from
- `n`: the number of groups to generate
- `sample_size`: the size of each group specified in `n`

Value

a dataframe containing the random samples with tidyverse code attached

Author(s)

Owen Jin

See Also

code

Examples

```
filtered <- filterRandom(iris, n = 5, sample_size = 3)
cat(code(filtered))
head(filtered)
```
**filterRows**  
*Filter data by row numbers*

**Description**

Filter a dataframe by slicing off specified rows and returns the result along with tidyverse code used to generate it.

**Usage**

```r
cat(filterRows(.data, rows))
```

**Arguments**

- `.data` a dataframe to filter
- `rows` a numeric vector of row numbers to slice off

**Value**

filtered dataframe with tidyverse code attached

**Author(s)**

Owen Jin

**See Also**

code

**Examples**

```r
filtered <- filterRows(iris, rows = c(1,4,5))
cat(code(filtered))
head(filtered)
```

---

**fitDesign**  
*Fit a survey design*

**Description**

Fit a survey design to an object

**Usage**

```r
fitDesign(svydes, dataset.name)
```
Arguments

svydes a design
dataset.name a dataset name

Value

a survey object

Author(s)

Tom Elliott

fitModel

Fit models

Description

Wrapper function for 'lm', 'glm', and 'svyglm'.

Usage

fitModel(
  y,
  x,
  data,
  family = "gaussian",
  link = switch(family, gaussian = "gaussian", binomial = "logit", poisson = "log"),
  design = "simple",
  svydes = NA,
  ...
)

Arguments

y character string representing the response,
x character string of the explanatory variables,
data name of the object containing the data.
family gaussian, binomial, poisson (so far, no others will be added)
link the link function to use
design data design specification. one of 'simple', 'survey' or 'experiment'
svydes a vector of arguments to be passed to the svydesign function, excluding data (defined above)
... further arguments to be passed to lm, glm, svyglm, such as offset, etc.
is_cat

Value
A model call formula (using lm, glm, or svyglm)

Author(s)
Tom Elliott

Description
The iNZightTools package contains a suite of helper functions for iNZight, mostly to make GUI development easier to provide some type of consistency across desktop and shiny versions.

Author(s)
Tom Elliott et al.

See Also
iNZight

is_cat

Description
This function checks if a variable a factor.

Usage
is_cat(x)

Arguments
x the variable to check

Value
logical, TRUE if the variable is a factor

Author(s)
Tom Elliott
is_dt

Is datetime check

Description
This function checks if a variable a date/time/datetime

Usage
is_dt(x)

Arguments
x the variable to check

Value
logical, TRUE if the variable is a datetime

Author(s)
Tom Elliott

is_num

Is numeric check

Description
This function checks if a variable is numeric, or could be considered one. For example, dates and times can be treated as numeric, so return TRUE.

Usage
is_num(x)

Arguments
x the variable to check

Value
logical, TRUE if the variable is numeric

Author(s)
Tom Elliott
is_preview

Description
Checks if the complete file was read or not.

Usage
is_preview(df)

Arguments
df    data to check

Value
logical

is_survey

Description
Check if object is a survey object (either standard or replicate design)

Usage
is_survey(x)

Arguments
x    object to be tested

Value
logical

Author(s)
Tom Elliott
is_svydesign

Check if object is a survey object (created by svydesign())

Description
Check if object is a survey object (created by svydesign())

Usage
is_svydesign(x)

Arguments
x object to be tested

Value
logical

Author(s)
Tom Elliott

is_svyrep

Check if object is a replicate survey object (created by svrepdesign())

Description
Check if object is a replicate survey object (created by svrepdesign())

Usage
is_svyrep(x)

Arguments
x object to be tested

Value
logical

Author(s)
Tom Elliott
### joindata

*Join data with another dataset*

**Description**

Join data with another dataset

**Usage**

```r
joindata(
  .data, imported_data, origin_join_col, import_join_col, join_method, left, right
)
```

**Arguments**

- `.data`: Original data
- `imported_data`: Imported dataset
- `origin_join_col`: Column selected from the original data
- `import_join_col`: Column selected from the imported dataset
- `join_method`: Function used to join the two datasets
- `left`: Suffix name assigned to the original dataset
- `right`: Suffix name assigned to the imported dataset

**Value**

Joined dataset

---

### load_rda

*Load object(s) from an Rdata file*

**Description**

Load object(s) from an Rdata file

**Usage**

```r
load_rda(file)
```
missingToCat

Arguments

file       path to an rdata file

Value

list of data frames, plus code

Author(s)

Tom Elliott

See Also

save_rda

Description

Turn <NA>'s into a "missing" character; hence numeric variables will be converted to categorical variables with any numeric values will be converted to "observed", and returns the result along with tidyverse code used to generate it.

Usage

missingToCat(.data, vars, names = paste0(vars, "_miss"))

Arguments

.data       a dataframe with the columns to convert its missing values into categorical
vars       a character vector of the variables in .data for conversion of missing values to categorical
names       a vector of names for the new variables

Value

original dataframe containing new columns of the converted variables for the missing values with tidyverse code attached

Author(s)

Owen Jin

See Also

code
Examples

```r
missing <- missingToCat(iris, vars = c("Species", "Sepal.Length"))
cat(code(missing))
head(missing)
```

---

**newdevice**

*Open a New Graphics Device*

**Description**

Opens a new graphics device

**Usage**

```r
newdevice(width = 7, height = 7, ...)
```

**Arguments**

- `width` the width (in inches) of the new device
- `height` the height (in inches) of the new device
- `...` additional arguments passed to the new device function

**Details**

Depending on the system, difference devices are better. The windows device works fine (for now), only attempt to speed up any other devices that we’re going to be using. We speed them up by getting rid of buffering.

**Author(s)**

Tom Elliott

---

**rankVars**

*Rank the data of a numeric variables*

**Description**

Rank the values of a numeric variable in descending order, and returns the result along with tidyverse code used to generate it. Ties are broken as such: eg. values = 5, 6, 6, 7 ; rank = 1, 2, 2, 3

**Usage**

```r
rankVars(.data, vars)
```
**Arguments**

- **.data**: a dataframe with the variables to rank
- **vars**: a character vector of numeric variables in `.data` to rank

**Value**

the original dataframe containing new columns with the ranks of the variables in `var` with tidyverse code attached

**Author(s)**

Owen Jin

**See Also**

- code

**Examples**

```r
ranked <- rankVars(iris, vars = c("Sepal.Length", "Petal.Length"))
cat(code(ranked))
head(ranked)
```

---

**Description**

This function will read a CSV file with iNZight metadata in the header. This allows plain text CSV files to be supplied with additional comments that describe the structure of the data to make import and data handling easier.

**Usage**

```r
read_meta(file, preview = FALSE, column_types, ...)
```

**Arguments**

- **file**: the plain text file with metadata
- **preview**: logical, if TRUE only the first 10 rows are returned
- **column_types**: optional column types
- **...**: more arguments

**Details**

The main example is to define factor levels for an integer variable in large data sets.
read_text

Read text as data

Description

The text can also be the value "clipboard" which will use `readr::clipboard()`.

Usage

```
read_text(txt, delim = '\t', ...)  
```

Arguments

- **txt** character string
- **delim** the delimiter to use, passed to `readr::read_delim()`
- **...** additional arguments passed to `readr::read_delim()`

Value

data.frame

Author(s)

Tom Elliott

renameLevels

Rename the levels of a categorical variable

Description

Rename the levels of a categorical variables, and returns the result along with tidyverse code used to generate it.

Usage

```
renameLevels(.data, var, to_be_renamed, name = sprintf("%s.rename", var))
```
renameVars

Arguments

.data a dataframe with the column to be renamed
var a character of the categorical variable to rename
to_be_renamed a list of the old level name assigned to the new level name; i.e., ‘list(’new level name’ = ’old level name’)’
name a name for the new variable

Value

original dataframe containing a new column of the renamed categorical variable with tidyverse code attached

Author(s)

Owen Jin

See Also
code

Examples

renamed <- renameLevels(iris, var = "Species",
                        to_be_renamed = list(set = "setosa", ver = "versicolor"))
cat(code(renamed))
head(renamed)

renameVars Rename column names

Description

Rename column names and returns the result along with tidyverse code used to generate it.

Usage

renameVars(.data, to_be_renamed_list)

Arguments

.data a dataframe with columns to rename
to_be_renamed_list a list of the new column names assigned to the old column names ie. list(’old column names’ = ’new column names’)
Value

original dataframe containing new columns of the renamed columns with tidyverse code attached

Author(s)

Owen Jin

See Also

code

Examples
code

renamed <- renameVars(iris,  
to_be_renamed_list = list(Species = "Type", Petal.Width = "P.W"))
cat(code(renamed))
head(renamed)

reorderLevels

Reorder a categorical

Description

Reorder the factors of a categorical variable either manually or frequency

Usage

reorderLevels(
  .data,  
  var,  
  new_levels = NULL,  
  freq = FALSE,  
  name = sprintf("%s.reord", var)
)

Arguments

.data a dataframe to reorder

var a categorical variable to reorder

new_levels a character vector of the new factor order. Only specify if freq = FALSE

freq logical, If freq = FALSE (default), will manually reorder using new_levels. If freq = TRUE, will reorder based of descending frequency of the factor levels

name name for the new variable
Value

original dataframe containing a new column of the reordered categorical variable with tidyverse code attached

Author(s)

Owen Jin

See Also

code

Examples

reordered <- reorderLevels(iris, var = "Species",
    new_levels = c("versicolor", "virginica", "setosa"))
cat(code(reordered))
head(reordered)

# reshape_data
Reshaping dataset from wide to long or from long to wide

Description

Reshaping dataset from wide to long or from long to wide

Usage

reshape_data(.data, col1, col2, cols, key, value, check)

Arguments

.data dataset
col1 column to spread out (for long to wide)
col2 values to be put in the spread out column (for long to wide)
cols columns(s) to gather together (for wide to long)
key name for new column containing old column names (for wide to long)
value name for new column containing old column values (for wide to long)
check check whether to use long to wide or wide to long

Value

reshaped dataset

Author(s)

Yiwen He
save_rda  
Save an object with, optionally, a (valid) name

Description
Save an object with, optionally, a (valid) name

Usage
save_rda(data, file, name)

Arguments
- data: the data frame to save
- file: where to save it
- name: optional, the name the data will have in the rda file

Value
logical, should be TRUE, along with code for the save

Author(s)
Tom Elliott

See Also
load_rda

separate  
Separate columns

Description
Separate columns

Usage
separate(.data, col, left, right, sep, check)
sheets

Arguments
.data dataset
col column to be separated
left name for the separated left column
right name for the separated right column
sep separator used to separate columns
check method of separating

Value
separated dataset

Author(s)
Yiwen He

Description
List of available sheets from a file

Usage
sheets(x)

Arguments
x a dataframe from smart_read

Value
vector of sheet names, or NULL

Author(s)
Tom Elliott
smart_read  

*iNZight Smart Read*

**Description**

A simple function that magically imports a file, irrespective of type.

**Usage**

```r
smart_read(
  file,
  ext = tools::file_ext(file),
  preview = FALSE,
  column_types = NULL,
  ...
)
```

**Arguments**

- `file`: the file path to read
- `ext`: file extension, namely "csv" or "txt"
- `preview`: logical, if TRUE only the first few rows of the data will be returned
- `column_types`: vector of column types (see `?readr::read_csv`)
- `...`: additional parameters passed to read_* functions

**Value**

a dataframe with attributes

**Author(s)**

Tom Elliott

sortVars  

*Sort data by variables*

**Description**

Sorts a dataframe by one or more variables, and returns the result along with tidyverse code used to generate it.

**Usage**

```r
sortVars(.data, vars, asc = rep(TRUE, length(vars)))
```
stackVars

Arguments

.data a dataframe to sort
vars a character vector of variable names to sort by
asc logical, same length as vars. If TRUE (default), sorted in ascending order, otherwise descending.

Value
data.frame with tidyverse code attached

Author(s)
Owen Jin

See Also
code

Examples
sorted <- sortVars(iris, vars = c("Sepal.Width", "Sepal.Length"),
asc = c(TRUE, FALSE))
cat(code(sorted))
head(sorted)

Description
Collapse columns by converting from a long format to a tall format and returns the result along with tidyverse code used to generate it.

Usage
stackVars(.data, vars, key = "stack.variable", value = "stack.value")

Arguments
.data a dataframe to stack
vars a character vector of variables to stack
key name of the new column for the stacked variables. "stack.variable" by default
value name of the new column for the stacked values of the stacked. "stack.value" by default
**Value**

stacked dataframe with tidyverse code attached

**Author(s)**

Owen Jin

**See Also**

code

**Examples**

```r
stacked <- stackVars(iris, vars = c("Species", "Sepal.Width"),
                      key = "Variable", value = "Value")
cat(code(stacked))
head(stacked)
```

---

**standardizeVars**  
*Standardize the data of a numeric variable*

**Description**

Centre then divide by the standard error of the values in a numeric variable

**Usage**

```r
standardizeVars(.data, vars, names = paste(sep = ".", vars, "std"))
```

**Arguments**

- `.data`: a dataframe with the columns to standardize
- `vars`: a character vector of the numeric variables in `.data` to standardize
- `names`: names for the created variables

**Value**

the original dataframe containing new columns of the standardized variables with tidyverse code attached

**Author(s)**

Owen Jin

**See Also**

code
Examples

```r
standardized <- standardizeVars(iris, var = c("Sepal.Width", "Petal.Width"))
cat(code(standardized))
head(standardized)
```

Description

Tidy code with correct indents and limit the code to the specific width

Usage

```r
tidy_all_code(x, width = 80, indent = 4, outfile, incl_library = TRUE)
```

Arguments

- `x` character string or file name of the file containing messy code
- `width` the width of a line
- `indent` how many spaces for one indent
- `outfile` the file name of the file containing formatted code
- `incl_library` logical, if true, the output code will contain library name

Value

formatted code, optionally written to `outfile`

Author(s)

Lushi Cai

Description

Transform the values of a numeric variable by applying a mathematical function
Usage

transformVar(
  .data,
  var,
  transformation,
  name = sprintf("%s.%s", transformation, var)
)

Arguments

.data a dataframe with the variables to transform
var a character of the numeric variable in .data to transform
transformation a name of a valid mathematical function that can be applied to numeric values, eg. "log", "exp", "sqrt". For squaring, use "square"; for inverting, use "reciprocal"
name the name of the new variable

Value

the original dataframe containing a new column of the transformed variable with tidyverse code attached

Author(s)

Owen Jin

See Also

code

Examples

transformed <- transformVar(iris, var = "Petal.Length",
  transformation = "log")
cat(code(transformed))
head(transformed)

unite Unite columns in a dataset

Description

Unite columns in a dataset

Usage

unite(.data, name, col, sep)
url_to_temp

Arguments
- .data: dataset
- name: name for the new united column
- col: a vector of column names
- sep: separator used in between the united columns

Value
- united dataset

Author(s)
- Yiwen He

url_to_temp

Description
- Download URL to temp file

Usage
- url_to_temp(url)

Arguments
- url: where the file lives on the internet

Value
- the location of a (temporary) file location

Author(s)
- Tom Elliott
validation_details  Details of Validation Rule Results

Description
Generates the more detailed text required for the details section in iNZValidateWin.

Usage
validation_details(cf, v, var, id.var, df)

Arguments
- cf: Confrontation object from validate::confront()
- v: Validator that generated cf
- var: Rule name to give details about
- id.var: Variable name denoting a unique identifier for each observation
- df: The dataset that was confronted

Value
A character vector giving each line of the summary detail text

Author(s)
Daniel Barnett

validation_summary  Validation Confrontation Summary

Description
Generates a summary of a confrontation which gives basic information about each validation rule tested.

Usage
validation_summary(cf)

Arguments
- cf: Confrontation object from validate::confront()
vartype

**Value**
A data.frame with number of tests performed, number of passes, number of failures, and failure percentage for each validation rule.

**Author(s)**
Daniel Barnett

---

| vartype | Get variable type name |

**Description**
Get variable type name

**Usage**

vartype(x)

**Arguments**

x vector to be examined

**Value**

character vector of the variable’s type

**Author(s)**
Tom Elliott
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>iNZight</td>
<td>18</td>
</tr>
<tr>
<td>iNZightTools</td>
<td>18</td>
</tr>
<tr>
<td>add_suffix</td>
<td>3</td>
</tr>
<tr>
<td>aggregateData</td>
<td>4, 10</td>
</tr>
<tr>
<td>aggregatedt</td>
<td>5</td>
</tr>
<tr>
<td>appendrows</td>
<td>5</td>
</tr>
<tr>
<td>code</td>
<td>4, 6, 7, 9, 11, 13–16, 23, 25, 27–29, 33, 34, 36</td>
</tr>
<tr>
<td>collapseLevels</td>
<td>7</td>
</tr>
<tr>
<td>combineCatVars</td>
<td>8</td>
</tr>
<tr>
<td>convert_to_datetime</td>
<td>9</td>
</tr>
<tr>
<td>convertToCat</td>
<td>8</td>
</tr>
<tr>
<td>countMissing</td>
<td>4, 10</td>
</tr>
<tr>
<td>create_varname</td>
<td>11</td>
</tr>
<tr>
<td>createNewVar</td>
<td>10</td>
</tr>
<tr>
<td>deleteVars</td>
<td>12</td>
</tr>
<tr>
<td>extract_part</td>
<td>12</td>
</tr>
<tr>
<td>filterLevels</td>
<td>13</td>
</tr>
<tr>
<td>filterNumeric</td>
<td>14</td>
</tr>
<tr>
<td>filterRandom</td>
<td>15</td>
</tr>
<tr>
<td>filterRows</td>
<td>16</td>
</tr>
<tr>
<td>fitDesign</td>
<td>16</td>
</tr>
<tr>
<td>fitModel</td>
<td>17</td>
</tr>
<tr>
<td>fitModel</td>
<td>17</td>
</tr>
<tr>
<td>iNZightTools</td>
<td>18</td>
</tr>
<tr>
<td>is_cat</td>
<td>18</td>
</tr>
<tr>
<td>is_dt</td>
<td>19</td>
</tr>
<tr>
<td>is_num</td>
<td>19</td>
</tr>
<tr>
<td>is_preview</td>
<td>20</td>
</tr>
<tr>
<td>is_survey</td>
<td>20</td>
</tr>
<tr>
<td>is_svydesign</td>
<td>21</td>
</tr>
<tr>
<td>is_svyrep</td>
<td>21</td>
</tr>
<tr>
<td>joindata</td>
<td>22</td>
</tr>
<tr>
<td>load_rda</td>
<td>22, 30</td>
</tr>
<tr>
<td>missingToCat</td>
<td>23</td>
</tr>
<tr>
<td>newdevice</td>
<td>24</td>
</tr>
<tr>
<td>rankVars</td>
<td>24</td>
</tr>
<tr>
<td>read_meta</td>
<td>25</td>
</tr>
<tr>
<td>read_text</td>
<td>26</td>
</tr>
<tr>
<td>renameLevels</td>
<td>26</td>
</tr>
<tr>
<td>renameVars</td>
<td>27</td>
</tr>
<tr>
<td>reorderLevels</td>
<td>28</td>
</tr>
<tr>
<td>reshape_data</td>
<td>29</td>
</tr>
<tr>
<td>save_rda</td>
<td>23, 30</td>
</tr>
<tr>
<td>separate</td>
<td>30</td>
</tr>
<tr>
<td>sheets</td>
<td>31</td>
</tr>
<tr>
<td>smart_read</td>
<td>32</td>
</tr>
<tr>
<td>sortVars</td>
<td>32</td>
</tr>
<tr>
<td>stackVars</td>
<td>33</td>
</tr>
<tr>
<td>standardizeVars</td>
<td>34</td>
</tr>
<tr>
<td>tidy_all_code</td>
<td>35</td>
</tr>
<tr>
<td>transformVar</td>
<td>35</td>
</tr>
<tr>
<td>unite</td>
<td>36</td>
</tr>
<tr>
<td>url_to_temp</td>
<td>37</td>
</tr>
<tr>
<td>validation_details</td>
<td>38</td>
</tr>
<tr>
<td>validation_summary</td>
<td>38</td>
</tr>
<tr>
<td>vartype</td>
<td>39</td>
</tr>
</tbody>
</table>