Package ‘tldr’

July 28, 2021

Title  T Loux Doing R: Functions to Simplify Data Analysis and Reporting
Version 0.2.3

Description  Gives a number of functions to aid common data analysis processes and reporting statistical results in an 'RMarkdown' file. Data analysis functions combine multiple base R functions used to describe simple bivariate relationships into a single, easy to use function. Reporting functions will return character strings to report p-values, confidence intervals, and hypothesis test and regression results. Strings will be LaTeX-formatted as necessary and will knit pretty in an 'RMarkdown' document. The package also provides a wrapper for the CreateTableOne() function in the 'tableone' package to make the results knit-able.

Depends  R (>= 3.3.1)
Imports  tableone, ggplot2, reshape2
Suggests  knitr
License GPL-3
Encoding UTF-8
LazyData false
RoxygenNote 7.1.1

R topics documented:
as_perc .......................................................... 2
cat_compare ..................................................... 3
cont_compare ..................................................... 4
cutp .............................................................. 4
as_perc

Description

as_perc formats a proportion as a percentage to print in an RMarkdown document.

Usage

as_perc(p, digits = 0)

Arguments

p A length-1 numeric to be interpreted as a proportion

digits Number of digits to round percentage to (default to 0)

Details

Simply multiplies p by 100 and affixes a percent sign to the end after rounding.

Value

Returns a string to report a percentage to the specified number of digits.

Examples

as_perc(0.2345)
as_perc(0.000234)
cat_compare

Investigate association between two categorical variables

Description

`cat_compare` gives details about the association between two categorical variables.

Usage

`cat_compare(x, y, plot = TRUE)`

Arguments

- `x`: A categorical variable: the predictor or group variable, if appropriate
- `y`: A categorical variable: the outcome, if appropriate
- `plot`: Logical. Whether a mosaic plot should be drawn

Details

Strictly, `x` and `y` do not need to be factors but will be coerced into factors.

Value

Returns a list including (1) two-way table of counts, (2) chi-squared test for independence, (3) Cramer’s V standardized effect, and (4) ggplot2 column plot of proportions conditional on `x`, if requested.

The table of counts will include missing values of both variables, but these rows/columns are discarded prior to the chi-squared test and Cramer’s V calculations.

Examples

```r
v1 = rbinom(n=50, size=1, p=0.5)
v2 = rbinom(n=50, size=2, p=0.3 + 0.2*v1)
cat_compare(x=v1, y=v2, plot=TRUE)
```
cont_compare  

*Compare a numerical variable across levels of a categorical variable*

**Description**

Deprecated. Use `num_compare` instead.

**Usage**

```r
cont_compare(y, grp, plot = c("density", "boxplot", "none"))
```

**Arguments**

- **y**: A numerical variable
- **grp**: A categorical variable
- **plot**: Type of plot to produce

**Value**

Returns a list including (1) group-wise summary statistics, (2) ANOVA decomposition, (3) eta-squared effect size, and (4) ggplot2 object, if requested.

cutp  

*Cut a numeric vector into quantiles*

**Description**

`cutp` is a wrapper for the base `cut` function. The vector `x` will be categorized using the percentiles provided in `p` to create break values.

**Usage**

```r
cutp(x, p, ...)```

**Arguments**

- **x**: A numeric vector to be discretized
- **p**: A numeric vector of probabilities
- **...**: Arguments passed to `cut`

**Details**

Within the `cutp` function, `p` is passed to `quantile` as the `probs` input. The computed quantiles are then used as the `breaks` in `cut`. The values `-Inf` and `Inf` are added to the beginning and end of the breaks vector, respectively, so quantiles for 0 and 1 do not need to be given explicitly.
**inline_coef**

**Value**

Returns the output from `cut`. This is usually a factor unless otherwise specified.

`# @seealso quantile; cut`

**Examples**

```r
myvals = rnorm(1000)
catx = cutp(x=myvals, p=c(0.25, 0.5, 0.75), labels=c('Q1', 'Q2', 'Q3', 'Q4'))
table(catx)
```

**inline_coef**

*Report the coefficient from a regression model inline*

**Description**

`inline_coef` presents the results of a coefficient from a `lm` or `glm` model in LaTeX format to be reported inline in an RMarkdown document.

**Usage**

```r
inline_coef(model, variable, coef = TRUE, stat = TRUE, pval = TRUE, digits = 2)
inline_coef_p(model, variable, digits = 2)
```

**Arguments**

- `model` A regression model
- `variable` A character string giving the name of the variable to be reported
- `coef` Logical, whether the coefficient value is to be reported (default TRUE)
- `stat` Logical, whether the test statistic for the coefficient should be reported (default TRUE)
- `pval` Logical, whether the p-value for the coefficient should be reported (default TRUE)
- `digits` Number of digits to round to (default to 2)

**Details**

This function currently only supports `lm` and `glm` objects. Suggestions and requests are welcomed.

`inline_coef_p` is a wrapper for `inline_coef` to report only the p-value (sets all non-p-value logicals to FALSE).

**Value**

Returns a LaTeX-formatted result for use in RMarkdown document.
Examples

```r
x1 = rnorm(20)
x2 = rnorm(20)
y = x1 + x2 + rnorm(20)
model1 = lm(y ~ x1 + x2)
inline_coef(model1, 'x1')
inline_coef_p(model1, 'x1')
```

```r
inline_reg(model, fit = TRUE, stat = TRUE, pval = TRUE, digits = 2)
inline_reg_p(model, digits = 2)
inline_anova(model, stat = TRUE, pval = TRUE, digits = 2)
```

Argument

- `model` A regression model
- `fit` Logical, whether the regression fit is to be reported (default TRUE, only applicable to `lm` objects)
- `stat` Logical, whether the test statistic for the coefficient should be reported (default TRUE)
- `pval` Logical, whether the p-value for the coefficient should be reported (default TRUE)
- `digits` Number of digits to round to (default to 2)

Details

For `lm` objects, results include R-squared, the F statistic, and the p-value. For `glm` objects, results include the chi-squared statistic and the p-value.

This function currently only supports `lm` and `glm` objects. Suggestions and requests are welcomed.

`inline_reg_p` is a wrapper for `inline_reg` to report only the p-value (sets all non-p-value logics to FALSE). `inline_anova` is a wrapper to report a one-way ANOVA result in which `fit` is set to FALSE and other logical inputs (`stat`, `pval`, and `digits`) are allowed to be user-defined.

Value

Returns a LaTeX-formatted result for use in RMarkdown document.
Examples

```r
x1 = rnorm(20)
y1 = x1 + rnorm(20)
model1 = lm(y1 ~ x1)
inline_reg(model1)
```

```r
x2 = rnorm(20)
y2 = rbinom(n=20, size=1, prob=pnorm(x2))
model2 = glm(y2 ~ x2, family=binomial('logit'))
inline_reg(model2)
```

---

**inline_test**

*Report a hypothesis test inline*

---

### Description

`inline_test` formats the results of an htest object into LaTeX to be presented inline in an RMarkdown document.

### Usage

```
inline_test(test, stat = TRUE, pval = TRUE, digits = 2)
inline_test_p(test, digits = 2)
```

### Arguments

- **test**: An htest object
- **stat**: Logical, whether to report test statistic (default TRUE)
- **pval**: Logical, whether to report p-value (default TRUE)
- **digits**: Number of digits to round to (default to 2)

### Details

This function currently only supports t tests and chi-squared tests. Suggestions and requests are welcomed.

`inline_test_p` is a wrapper for `inline_test` to report only the p-value (sets all non-p-value logics to FALSE).

### Value

Returns a LaTeX-formatted hypothesis test result for use in RMarkdown document.
Examples

```r
x = rnorm(20)
test1 = t.test(x)
inline_test(test1)
inline_test_p(test1)
```

---

**KreateTableOne**  
*Create a table of descriptive statistics formatted for knitr::kable*

---

**Description**

*KreateTableOne* is a wrapper for *tableone::CreateTableOne* which formats the original plain text table as a data.frame of character columns. This can be printed in an RMarkdown document in a number of ways, e.g., using `knitr::kable`. *svyKreateTabeOne* does the same with *tableone::svyCreateTableOne*.

**Usage**

```r
KreateTableOne(x, printSMD = TRUE, ...)
svyKreateTableOne(x, ..., printSMD = TRUE)
```

**Arguments**

- `x`  
  The data set to be passed to the `data` parameter of *tableone::CreateTableOne*
- `printSMD`  
  Logical passed to `smd` parameter of *tableone::CreateTableOne*
- `...`  
  Other parameters to be passed to *tableone::CreateTableOne*

**Details**

This is a very hacky function. If used within an RMarkdown document, *KreateTableOne* should be called in a code chunk with `results='hide'` to hide the plain test results printed from *tableone::CreateTableOne*. The resulting data frame should be saved as an object and used in a second code chunk for formatted printing. Suggestions for improvement are welcomed.

The function is written to work with `knitr::kable`, but should be able to work with other functions such as `xtable::xtable`.

**Value**

Returns a data frame of character columns.

**See Also**

*CreateTableOne*
Examples

table1 = KreateTableOne(x=mtcars, strata='am', factorVars='vs')
table1
knitr::kable(table1)

---

**num_compare**

*Compare a numerical variable across levels of a categorical variable*

Description

`num_compare` gives details about the distribution of a numeric variable across subsets of the dataset

Usage

```r
num_compare(y, grp, plot = c("density", "boxplot", "none"))
```

Arguments

- `y`: A numerical variable
- `grp`: A categorical variable
- `plot`: Type of plot to produce

Value

Returns a list including (1) group-wise summary statistics, (2) ANOVA decomposition, (3) eta-squared effect size, and (4) ggplot2 object, if requested.

Examples

```r
v1 = rbinom(n=50, size=1, p=0.5)
v2 = rnorm(50)
num_compare(y=v2, grp=v1, plot='density')
```
write_int  
*Format an interval for display*

**Description**

write_int formats a numeric input into an interval to be printed, e.g., in an RMarkdown document.

**Usage**

```r
write_int(x, delim = "(", digits = 2)
```

**Arguments**

- `x` A length-2 numeric vector consisting of the endpoints of the interval or an n-row by 2-column matrix of endpoints.
- `delim` The bracket delimiters to surround the interval. Must be either a round bracket, square bracket, curly bracket, or angled bracket.
- `digits` Number of digits to round to (default to 2). Will keep trailing zeros.

**Details**

If a matrix is provided, the values in each row will be used to create a formatted interval.

**Value**

Returns a character string of the form "(x[1], x[2])" (or supplied bracket delimiter).

**Examples**

```r
write_int(x=c(1.2, 2.345))
write_int(x=c(1.2, 2.345), delim='[')
```

---

write_p  
*Format a p-value for display*

**Description**

write_p formats a p-value for display in an RMarkdown document.

**Usage**

```r
write_p(x, digits = 2)
```
Arguments

- **x**: A length-1 numeric or a list-like object with element named `p.value` (such as an `htest` object)
- **digits**: Number of digits to round to (default to 2)

Details

If \( x < 10^{-\text{digits}} \), then the result is the string \( p < 10^{-\text{digits}} \) in decimal notation.

Value

Returns a LaTeX-formatted string to report a p-value to the specified number of digits.

Examples

```r
write_p(0.2345)

write_p(0.000234)

x = rnorm(10)
test1 = t.test(x)
write_p(test1)
```
Index

as_perc, 2
cat_compare, 3
cont_compare, 4
CreateTableOne, 8
cut, 5
cutp, 4
inline_anova (inline_reg), 6
inline_coef, 5
inline_coef_p (inline_coef), 5
inline_reg, 6
inline_reg_p (inline_reg), 6
inline_test, 7
inline_test_p (inline_test), 7
KreateTableOne, 8
num_compare, 4, 9
quantile, 5
svyKreateTableOne (KreateTableOne), 8
write_int, 10
write_p, 10