Package ‘roistats’

March 10, 2021

Title  Convenience Functions for Applying Basic Stats to Sub-Groups
Version  0.1.1
Description  Easily applying same t-tests/basic data description across several sub-
groups, with the output as a nice arranged data.frame. Multiple comparison and the signifi-
cance symbols are also provided.
License  MIT + file LICENSE
Depends  R (>= 3.5.0)
URL  https://github.com/Irisfee/roistats
BugReports  https://github.com/Irisfee/roistats/issues
Encoding  UTF-8
LazyData  true
RoxygenNote  7.1.1
Imports  dplyr, tidyr, magrittr, purrr, stats, rlang
Suggests  testthat (>= 3.0.0), covr, knitr, rmarkdown
Config/testthat/edition  3
VignetteBuilder  knitr
NeedsCompilation  no
Author  Yufei Zhao [aut, cre] (<https://orcid.org/0000-0002-6511-8507>),
        Daniel Anderson [ctb] (<https://orcid.org/0000-0003-4699-4680>)
Maintainer  Yufei Zhao <fay.zz0213@gmail.com>
Repository  CRAN
Date/Publication  2021-03-10 17:20:02 UTC

R topics documented:

color_index ................................................................. 2
color_index_two_sample .............................................. 2
df_sem ................................................................. 3
p_range ................................................................. 4
t_test_one_sample .................................................. 4
t_test_two_sample ................................................... 5
## Index

<table>
<thead>
<tr>
<th>color_index</th>
<th>Color index</th>
</tr>
</thead>
</table>

### Description

The pre-processed data for identifying which brain regions is sensitive to the color memory of learned objects.

### Usage

```r
color_index
```

### Format

A tibble with 232 rows and 3 variables with one group attribute:

- **subj_id**: Subject identity number
- **roi_id**: Brain sub-region that of interest for the analysis. Served as the grouping variable here.
- **color_index**: A value we want to test if it is significantly different from 0 across subjects.

### References


### Examples

```r
color_index
```

---

## color_index_two_sample

### Description

The pre-processed data for identifying which brain regions is sensitive to the color memory of learned objects.

### Usage

```r
color_index_two_sample
```

### Description

The pre-processed data for identifying which brain regions is sensitive to the color memory of learned objects.

### Usage

```r
color_index_two_sample
```
Format

A tibble with 464 rows and 4 variables with one group attribute:

- **subj_id**: Subject identity number
- **roi_id**: Brain sub-region that of interest for the analysis. Served as the grouping variable here.
- **group**: A within-group variable for each subject. Indicating whether the color effect value is for the Paired or Control condition
- **color_effect**: A value we want to test between the two groups (Paired vs Control).

References


Examples

```r
color_index_two_sample
```

---

**Description**

Generate standard error of mean

**Usage**

```r
df_sem(data, x)
```

**Arguments**

- **data**: A data frame, generally grouped by the intended sub-groups which you want to compare for the same t-test.
- **x**: A (bare) column name of the variable which you want to get the mean, sd, and standard error of the mean (SEM).

**Value**

A data frame with consisting of characters. The columns that are always present are: group variable(s), mean, sd, n, and se(SEM).

**Examples**

```r
df_sem(color_index, color_index)
```

library(magrittr)
color_index %>%
  df_sem(color_index)
### p_range

Create significant symbols for p-values

**Description**

Create significant symbols for p-values

**Usage**

```r
p_range(p)
```

**Arguments**

- `p` A numeric p value (usually yielded from a statistical test).

**Value**

A character significant symbol. * represents the p is within the range of (0.05, 0.01], ** for (0.01, 0.001], and *** for (0.001, +inf]

**Examples**

```r
p_range(0.02)
library(dplyr)
t_test_one_sample(color_index, "color_index", mu = 0) %>% mutate(sig = p_range(p))
```

---

### t_test_one_sample

Generate one-sample t-test results for multiple sub-groups

**Description**

This function produce one-sample t-test (two-tailed with confident interval at 0.95) results for multiple sub-groups and provides with a nice output in a table format. It can also add adjusted p values for multiple comparison issue.

**Usage**

```r
t_test_one_sample(data, x, mu = 0, p_adjust = "bonferroni")
```
**t_test_two_sample**

**Arguments**

- **data**
  A grouped data frame. It should be grouped by the intended sub-groups which you want to do the same t-test.

- **x**
  Column name of the variable which contains data values that you want to test (see t.test and details).

- **mu**
  A number indicating the true value of the mean (or difference in means if you are performing a two sample test).

- **p_adjust**
  Character indicating which method should be used for adjusting multiple comparisons (see p.adjust and details). The default "bonferroni" corresponds to Bonferroni adjustment.

**Value**

A data.frame with the t-statistics table consisting of characters. The columns that are always present are: group variable(s), tvalue, df (degrees of freedom), p, and p_adjustmethod(s).

**Examples**

```r
# use bonferroni and fdr method for adjusted p values.
t_test_one_sample("color_index", mu = 0, p_adjust = c("bonferroni","fdr"))
```

---

**Description**

This function produce two-sample t-test (two-tailed with confident interval at 0.95) results for multiple sub-groups and provides with a nice output in a table format. It can also add adjusted p values for multiple comparison issue.

**Usage**

```r
t_test_two_sample(data, x, y, paired = FALSE, p_adjust = "bonferroni")
```
paired a logical indicating whether you want a paired t-test.
p_adjust character indicating which method should be used for adjusting multiple comparisons (see p.adjust and details). The default "bonferroni" corresponds to Bonferroni adjustment.

Value
A data.frame with the t-statistics table consisting of characters. The columns that are always present are: group variable(s), tvalue, df (degrees of freedom), p, and p_adjustmethod(s).

Examples
t_test_two_sample(color_index_two_sample, x = "color_effect", y = "group", paired = TRUE)

# use bonferroni and fdr method for adjusted p values.
library(magrittr)
color_index_two_sample %>%
t_test_two_sample(
  x = "color_effect", y = "group", paired = TRUE, p_adjust = c("bonferroni","fdr")
)
Index

* datasets
  color_index, 2
  color_index_two_sample, 2

color_index, 2
color_index_two_sample, 2

df_sem, 3
p.adjust, 5, 6
p_range, 4

t.test, 5
t_test_one_sample, 4
t_test_two_sample, 5