Package ‘catfun’

October 12, 2022

Type Package

Title Categorical Data Analysis

Version 0.1.4

Author Nick Williams

Maintainer Nick Williams <ntwilliams.personal@gmail.com>

Description Includes wrapper functions around existing functions for the analysis of categorical data and introduces functions for calculating risk differences and matched odds ratios. R currently supports a wide variety of tools for the analysis of categorical data. However, many functions are spread across a variety of packages with differing syntax and poor compatibility with each another. prop_test() combines the functions binom.test(), prop.test() and BinomCI() into one output. prop_power() allows for power and sample size calculations for both balanced and unbalanced designs. riskdiff() is used for calculating risk differences and matched_or() is used for calculating matched odds ratios. For further information on methods used that are not documented in other packages see Nathan Mantel and William Haenszel (1959) <doi:10.1093/jnci/22.4.719> and Alan Agresti (2002) <ISBN:0-471-36093-7>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports epitools, DescTools, cli, magrittr, Hmisc, broom, rlang

RoxygenNote 6.1.1

Suggests testthat, dplyr, forcats

NeedsCompilation no

Repository CRAN

Date/Publication 2019-06-14 14:10:03 UTC

R topics documented:

  matched_or ................................................................. 2
  matched_or.data.frame .................................................. 3
  matched_or.table ....................................................... 4
matched_or

Matched pairs odds ratio and confidence interval

Description

Create odds ratio and confidence interval from matched pairs data.

Usage

matched_or(df, ...)

Arguments

df

a dataframe with binary variables x and y or a 2 x 2 frequency table/matrix. If a table or matrix, x and y must be NULL. Used to select method.

... further arguments passed to or from other methods.

Details

The matched pairs odds ratio and confidence interval is the equivalent of calculating a Cochran-Mantel-Haenszel odds ratio where each pair is treated as a stratum.

Value

a list with class "matched_or" with the following components:

tab 2x2 table using for calculating risk difference

or dataframe with columns corresponding to matched-pairs OR, lower bound, and upper bound of CI

conf.level specified confidence level
Examples

```r
set.seed(1)
gene <- data.frame(pair = seq(1:35),
                   ulcer = rbinom(35, 1, .7),
                   healthy = rbinom(35, 1, .4))

matched_or(gene, ulcer, healthy)
```

---

**matched_or.data.frame**  
*Matched pairs odds ratio from a data frame*

**Description**

Create odds ratio and confidence interval from matched pairs data.

**Usage**

```r
## S3 method for class 'data.frame'
matched_or(df, x, y, weight = NULL, alpha = 0.05,
            rev = c("neither", "rows", "columns", "both"), ...)
```

**Arguments**

- `df` a dataframe with binary variables `x` and `y`.
- `x` binary vector, used as rows for frequency table and calculations.
- `y` binary vector, used as columns for frequency table and calculations.
- `weight` an optional vector of count weights.
- `alpha` level of significance for confidence interval.
- `rev` reverse order of cells. Options are "row", "columns", "both", and "neither" (default).
- `...` further arguments passed to or from other methods.

**Value**

a list with class "matched_or" with the following components:

- `tab` 2x2 table using for calculating risk difference
- `or` dataframe with columns corresponding to matched-pairs OR, lower bound, and upper bound of CI
- `conf.level` specified confidence level
Examples

gene <- data.frame(pair = seq(1:35),
                   ulcer = rbinom(35, 1, .7),
                   healthy = rbinom(35, 1, .4))

matched_or(gene, ulcer, healthy)

matched_or.table

Matched pairs odds ratio from a table

Description

Create odds ratio and confidence interval from matched pairs data.

Usage

```r
## S3 method for class 'table'
matched_or(df, alpha = 0.05, rev = c("neither", "rows", "columns", "both"), ...)
```

Arguments

df a dataframe with binary variables x and y or a 2 x 2 frequency table/matrix.
alpha level of significance for confidence interval.
rev reverse order of cells. Options are "row", "columns", "both", and "neither" (default).
... further arguments passed to or from other methods.

Value

a list with class "matched_or" with the following components:

- `tab` 2x2 table using for calculating risk difference
- `or` dataframe with columns corresponding to matched-pairs OR, lower bound, and upper bound of CI
- `conf.level` specified confidence level

Examples

```r
gene <- data.frame(pair = seq(1:35),
                   ulcer = rbinom(35, 1, .7),
                   healthy = rbinom(35, 1, .4))

gene_tab <- xtabs(~ ulcer + healthy, data = gene)
gene_tab %>% matched_or()
```
prop_power

---

**prop_power**  
*Power and sample size for 2 proportions*

---

**Description**

Calculate power and sample size for comparison of 2 proportions for both balanced and unbalanced designs.

**Usage**

```r
prop_power(n, n1, n2, p1, p2, fraction = 0.5, alpha = 0.05,  
          power = NULL, alternative = c("two.sided", "one.sided"), odds.ratio,  
          percent.reduction, ...)```

**Arguments**

- `n` total sample size.
- `n1` sample size in group 1.
- `n2` sample size in group 2.
- `p1` group 1 proportion.
- `p2` group 2 proportion.
- `fraction` fraction of total observations that are in group 1.
- `alpha` significance level/type 1 error rate.
- `power` desired power, between 0 and 1.
- `alternative` alternative hypothesis, one- or two-sided test.
- `odds.ratio` odds ratio comparing p2 to p1.
- `percent.reduction` percent reduction of p1 to p2.
- `...` further arguments passed to or from other methods.

**Details**

Power calculations are done using the methods described in `stats::power.prop.test`, `Hmisc::bsamsize`, and `Hmisc::bpower`.

**Value**

A list with class "prop_power" containing the following components:

- `n` the total sample size
- `n1` the sample size in group 1
- `n2` the sample size in group 2
- `p1` the proportion in group 1
- `p2` the proportion in group 2
- `power` calculated or desired power
- `sig.level` level of significance
prop_test

See Also
[stats::power.prop.test], [Hmisc::bsamsize], [Hmisc::bpower]

Examples
prop_power(n = 220, p1 = 0.35, p2 = 0.2)
prop_power(p1 = 0.35, p2 = 0.2, fraction = 2/3, power = 0.85)
prop_power(p1 = 0.35, n = 220, percent.reduction = 42.857)
prop_power(p1 = 0.35, n = 220, odds.ratio = 0.4642857)

prop_test

Tests for equality of proportions

Description
Conduct 1-sample tests of proportions and tests for equality of k proportions.

Usage
prop_test(x, ...)

Arguments
x
a vector of counts, a one-dimensional table with two entries, or a two-dimensional
table with 2 columns. Used to select method.

... further arguments passed to or from other methods.

Details
Calculations are done using the methods described in `stats::binom.test()` and `stats::prop.test()`

Value
a list with class "prop_test" containing the following components:

x number of successes
n number of trials
p null proportion
statistic the value of Pearson’s chi-squared test statistic
p_value p-value corresponding to chi-squared test statistic
df degrees of freedom
method the method used to calculate the confidence interval
method_ci confidence interval calculated using specified method
exact_ci exact confidence interval
exact_p p-value from exact test
prop_test.data.frame

Tests for equality of proportions

Description

Conduct 1-sample tests of proportions and tests for equality of k proportions.

Usage

## S3 method for class 'data.frame'
prop_test(x, pred, out, weight = NULL, rev = c("neither", "rows", "columns", "both"), method = c("wald", "wilson", "agresti-couli", "jeffreys", "modified wilson", "wilsoncc", "modified jeffreys", "clopper-pearson", "arcsine", "logit", "witting", "pratt"), alternative = c("two.sided", "less", "greater"), conf.level = 0.95, correct = FALSE, exact = FALSE, ...)

Arguments

x a dataframe with categorical variable pred and binary outcome out.
pred predictor/exposure, vector.
out outcome, vector.
weight an optional vector of count weights.
rev reverse order of cells. Options are "row", "columns", "both", and "neither" (default).

See Also

[stats::binom.test()], [stats::prop.test()]

Examples

prop_test(7, 50, method = "wald", p = 0.2)
prop_test(7, 50, method = "wald", p = 0.2, exact = TRUE)
prop_test(c(23, 24), c(50, 55))

vietnam <- data.frame(
  service = c(rep("yes", 2), rep("no", 2)),
  sleep = c(rep(c("yes", "no"), 2)),
  count = c(173, 160, 599, 851)
)

sleep <- xtabs(count ~ service + sleep, data = vietnam)
prop_test(sleep)
prop_test(vietnam, service, sleep, count)
**method** a character string indicating method for calculating confidence interval, default is "wald". Options include, wald, wilson, agresti-couli, jeffreys, modified wilson, wilsoncc modified jeffreys, clopper-pearson, arcsine, logit, witting, and pratt.

**alternative** character string specifying the alternative hypothesis. Possible options are "two.sided" (default), "greater", or "less".

**conf.level** confidence level for confidence interval, default is 0.95.

**correct** a logical indicating whether Yate’s continuity correction should be applied.

**exact** a logical indicating whether to output exact p-value, ignored if k-sample test.

**...** further arguments passed to or from other methods.

**Value**

a list with class "prop_test" containing the following components:

- **x** number of successes
- **n** number of trials
- **p** null proportion
- **statistic** the value of Pearson’s chi-squared test statistic
- **p_value** p-value corresponding to chi-squared test statistic
- **df** degrees of freedom
- **method** the method used to calculate the confidence interval
- **method_ci** confidence interval calculated using specified method
- **exact_ci** exact confidence interval
- **exact_p** p-value from exact test

**Examples**

```r
vietnam <- data.frame(
  service = c(rep("yes", 2), rep("no", 2)),
  sleep = c(rep(c("yes", "no"), 2)),
  count = c(173, 160, 599, 851)
)

prop_test(vietnam, service, sleep, count)
```
prop_test.matrix

Tests for equality of proportions

Description
Conduct 1-sample tests of proportions and tests for equality of k proportions.

Usage

```r
## S3 method for class 'matrix'
prop_test(x, method = c("wald", "wilson", "agresti-couli", "jeffreys", "modified wilson", "wilsoncc", "modified jeffreys", "clopper-pearson", "arcsine", "logit", "witting", "pratt"), alternative = c("two.sided", "less", "greater"), conf.level = 0.95, correct = FALSE, exact = FALSE, ...)
```

Arguments

- `x`: a 2 x k matrix.
- `method`: a character string indicating method for calculating confidence interval, default is "wald". Options include, wald, agresti-couli, jeffreys, modified wilson, wilsoncc, modified jeffreys, clopper-pearson, arcsine, logit, witting, and pratt.
- `alternative`: character string specifying the alternative hypothesis. Possible options are "two.sided" (default), "greater", or "less".
- `conf.level`: confidence level for confidence interval, default is 0.95.
- `correct`: a logical indicating whether Yate's continuity correction should be applied.
- `exact`: a logical indicating whether to output exact p-value, ignored if k-sample test.
- `...`: further arguments passed to or from other methods.

Value

a list with class "prop_test" containing the following components:

- `x`: number of successes
- `n`: number of trials
- `p`: null proportion
- `statistic`: the value of Pearson's chi-squared test statistic
- `p_value`: p-value corresponding to chi-squared test statistic
- `df`: degrees of freedom
- `method`: the method used to calculate the confidence interval
- `method_ci`: confidence interval calculated using specified method
- `exact_ci`: exact confidence interval
- `exact_p`: p-value from exact test
Examples

```r
matrix(c(23, 48, 76, 88), nrow = 2, ncol = 2) %>% prop_test()
```

Description

Conduct 1-sample tests of proportions and tests for equality of k proportions.

Usage

```r
## S3 method for class 'numeric'
prop_test(x, n, p = 0.5, method = c("wald", "wilson", "agresti-couli", "jeffreys", "modified wilson", "wilsoncc", "modified jeffreys", "clopper-pearson", "arcsine", "logit", "witting", "pratt"), alternative = c("two.sided", "less", "greater"), conf.level = 0.95, correct = FALSE, exact = FALSE, ...)
```

Arguments

- `x`: a vector of counts.
- `n`: a vector of counts of trials.
- `p`: a probability for the null hypothesis when testing a single proportion; ignored if comparing multiple proportions.
- `method`: a character string indicating method for calculating confidence interval, default is "wald". Options include, wald, wilson, agresti-couli, jeffreys, modified wilson, wilsoncc modified jeffreys, clopper-pearson, arcsine, logit, witting, and pratt.
- `alternative`: character string specifying the alternative hypothesis. Possible options are "two.sided" (default), "greater", or "less".
- `conf.level`: confidence level for confidence interval, default is 0.95.
- `correct`: a logical indicating whether Yate’s continuity correction should be applied.
- `exact`: a logical indicating whether to output exact p-value, ignored if k-sample test.
- `...`: further arguments passed to or from other methods.

Value

A list with class "prop_test" containing the following components:

- `x`: number of successes
- `n`: number of trials
- `p`: null proportion
- `statistic`: the value of Pearson’s chi-squared test statistic
prop_test.table

<table>
<thead>
<tr>
<th>p_value</th>
<th>p-value corresponding to chi-squared test statistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>df</td>
<td>degrees of freedom</td>
</tr>
<tr>
<td>method</td>
<td>the method used to calculate the confidence interval</td>
</tr>
<tr>
<td>method_ci</td>
<td>confidence interval calculated using specified method</td>
</tr>
<tr>
<td>exact_ci</td>
<td>exact confidence interval</td>
</tr>
<tr>
<td>exact_p</td>
<td>p-value from exact test</td>
</tr>
</tbody>
</table>

Examples

prop_test(7, 50, method = "wald", p = 0.2)
prop_test(7, 50, method = "wald", p = 0.2, exact = TRUE)

prop_test.table Tests for equality of proportions

Description

Conduct 1-sample tests of proportions and tests for equality of k proportions.

Usage

## S3 method for class 'table'
prop_test(x, method = c("wald", "wilson", "agresti-couli", "jeffreys", "modified wilson", "wilsoncc", "modified jeffreys", "clopper-pearson", "arcsine", "logit", "witting", "pratt"), alternative = c("two.sided", "less", "greater"), conf.level = 0.95, correct = FALSE, exact = FALSE, ...)

Arguments

x a 2 x k table.
method a character string indicating method for calculating confidence interval, default is "wald". Options include, wald, wilson, agresti-couli, jeffreys, modified wilson, wilsoncc modified jeffreys, clopper-pearson, arcsine, logit, witting, and pratt.
alternative character string specifying the alternative hypothesis. Possible options are "two.sided" (default), "greater", or "less".
conf.level confidence level for confidence interval, default is 0.95.
correct a logical indicating whether Yate's continuity correction should be applied.
extact a logical indicating whether to output exact p-value, ignored if k-sample test.
... further arguments passed to or from other methods.
Value

a list with class "prop_test" containing the following components:

- **x** number of successes
- **n** number of trials
- **p** null proportion
- **statistic** the value of Pearson’s chi-squared test statistic
- **p_value** p-value corresponding to chi-squared test statistic
- **df** degrees of freedom
- **method** the method used to calculate the confidence interval
- **method_ci** confidence interval calculated using specified method
- **exact_ci** exact confidence interval
- **exact_p** p-value from exact test

Examples

```r
data(vietnam)
xtabs(count ~ service + sleep, data = vietnam) %>% prop_test()
```

---

### riskdiff

**Risk difference**

Calculate risk difference and 95 percent confidence interval using Wald method.

**Usage**

```r
riskdiff(df, ...)
```

**Arguments**

- **df** a dataframe with binary variables `x` and `y` or a $2 \times 2$ frequency table/matrix. If a table or matrix, `x` and `y` must be NULL. Used to select method.
- **...** further arguments passed to or from other methods.
Value

a list with class "rdiff" containing the following components:

- rd: risk difference
- conf.level: specified confidence level
- ci: calculated confidence interval
- p1: proportion one
- p2: proportion two
- tab: 2x2 table using for calculating risk difference

Examples

```r
trial <- data.frame(
  disease = c(rep("yes", 2), rep("no", 2)),
  treatment = c(rep(c("estrogen", "placebo"), 2)),
  count = c(751, 623, 7755, 7479))

riskdiff(trial, treatment, disease, count, rev = "columns")
```

Description

Calculate risk difference and 95 percent confidence interval using Wald method.

Usage

```r
## S3 method for class 'data.frame'

riskdiff(df, x = NULL, y = NULL, weight = NULL, conf.level = 0.95, rev = c("neither", "rows", "columns", "both"), ...)
```

Arguments

df: a dataframe with binary variables x and y.

x: binary predictor/exposure, vector.
y: binary outcome, vector.

weight: an optional vector of count weights.

conf.level: confidence level for confidence interval, default is 0.95.

rev: reverse order of cells. Options are "row", "columns", "both", and "neither" (default).

...: further arguments passed to or from other methods.
riskdiff.matrix

Value

a list with class "rdiff" containing the following components:

- **rd**: risk difference
- **conf.level**: specified confidence level
- **ci**: calculated confidence interval
- **p1**: proportion one
- **p2**: proportion two
- **tab**: 2x2 table using for calculating risk difference

Examples

```r
trial <- data.frame(
  disease = c(rep("yes", 2), rep("no", 2)),
  treatment = c(rep(c("estrogen", "placebo"), 2)),
  count = c(751, 623, 7755, 7479))

riskdiff(trial, treatment, disease, count, rev = "columns")
```

<table>
<thead>
<tr>
<th>riskdiff.matrix</th>
<th>Risk difference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Description

Calculate risk difference and 95 percent confidence interval using Wald method.

Usage

```r
## S3 method for class 'matrix'
riskdiff(df, conf.level = 0.95, dnn = NULL,
  rev = c("neither", "rows", "columns", "both"), ...)```

Arguments

df
  a 2 x 2 frequency matrix.

conf.level
  confidence level for confidence interval, default is 0.95.

dnn
  optional character vector of dimension names.

rev
  reverse order of cells. Options are "row", "columns", "both", and "neither" (default).

... further arguments passed to or from other methods.
riskdiff.table

Value

a list with class "rdiff" containing the following components:

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rd</td>
<td>risk difference</td>
</tr>
<tr>
<td>conf.level</td>
<td>specified confidence level</td>
</tr>
<tr>
<td>ci</td>
<td>calculated confidence interval</td>
</tr>
<tr>
<td>p1</td>
<td>proportion one</td>
</tr>
<tr>
<td>p2</td>
<td>proportion two</td>
</tr>
<tr>
<td>tab</td>
<td>2x2 table using for calculating risk difference</td>
</tr>
</tbody>
</table>

Examples

```r
matrix(c(12, 45, 69, 15), nrow = 2, ncol = 2) %>%
riskdiff(dnn = c("New Drug", "Adverse Outcome"))
```

Description

Calculate risk difference and 95 percent confidence interval using Wald method.

Usage

```r
## S3 method for class 'table'
riskdiff(df, conf.level = 0.95, rev = c("neither", 
  "rows", "columns", "both"), ...)
```

Arguments

- `df`: a 2 x 2 frequency table.
- `conf.level`: confidence level for confidence interval, default is 0.95.
- `rev`: reverse order of cells. Options are "row", "columns", "both", and "neither" (default).
- `...`: further arguments passed to or from other methods.

Value

a list with class "rdiff" containing the following components:

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rd</td>
<td>risk difference</td>
</tr>
<tr>
<td>conf.level</td>
<td>specified confidence level</td>
</tr>
<tr>
<td>ci</td>
<td>calculated confidence interval</td>
</tr>
<tr>
<td>p1</td>
<td>proportion one</td>
</tr>
<tr>
<td>p2</td>
<td>proportion two</td>
</tr>
<tr>
<td>tab</td>
<td>2x2 table using for calculating risk difference</td>
</tr>
</tbody>
</table>
Examples

```r
trial <- data.frame(
  disease = c(rep("yes", 2), rep("no", 2)),
  treatment = c(rep(c("estrogen", "placebo"), 2)),
  count = c(751, 623, 7755, 7479))

xtabs(count ~ treatment + disease, data = trial) %>% riskdiff()
```

---

`tavolo` *Create 2 x k frequency tables*

**Description**

Helper function for creating 2 x k frequency tables.

**Usage**

`tavolo(df, ...)`

**Arguments**

- `df`  
  a dataframe with binary variable y and categorical variable x or a 2 x k frequency table/matrix. If a table or matrix, x and y must be NULL. Used to select method.
- `...`  
  further arguments passed to or from other methods.

**Value**

- `tab`  
  2 x k frequency table

**Examples**

```r
trial <- data.frame(disease = c(rep("yes", 2), rep("no", 2)),
  treatment = c(rep(c("estrogen", "placebo"), 2)),
  count = c(751, 623, 7755, 7479))

tavolo(trial, treatment, disease, count)
```
tavolo.data.frame  

Create 2 x k frequency tables

Description

Helper function for creating 2 x k frequency tables.

Usage

## S3 method for class 'data.frame'
tavolo(df, x, y, weight = NULL, rev = c("neither", "rows", "columns", "both"), ...)

Arguments

df  
a dataframe with binary variable y and categorical variable x.

x  
categorical predictor/exposure, vector.

y  
binary outcome, vector.

weight  
an optional vector of count weights.

rev  
character string indicating whether to switch row or column order, possible options are "neither", "rows", "columns", or "both". The default is "neither".

...  
further arguments passed to or from other methods.

Value

tab  
2 x k frequency table

Examples

trial <- data.frame(disease = c(rep("yes", 2), rep("no", 2)),
                     treatment = c(rep(c("estrogen", "placebo"), 2)),
                     count = c(751, 623, 7755, 7479))

tavolo(trial, treatment, disease, count)

---

tavolo.matrix  

Create 2 x k frequency tables

Description

Helper function for creating 2 x k frequency tables.
## S3 method for class 'matrix'
tavolo(df, dnn = NULL, rev = c("neither", "rows", "columns", "both"), ...)

### Arguments
- **df**: a 2 x k frequency matrix.
- **dnn**: optional character vector of dimension names.
- **rev**: character string indicating whether to switch row or column order, possible options are "neither", "rows", "columns", or "both". The default is "neither".
- **...**: further arguments passed to or from other methods.

### Value
- **tab**: 2 x k frequency table

### Examples
```r
tavolo(matrix(c(23, 45, 67, 12), nrow = 2, ncol = 2), rev = "both")
```

---

### Create 2 x k frequency tables

**tavolo.table**

Helper function for creating 2 x k frequency tables.

## S3 method for class 'table'
tavolo(df, rev = c("neither", "rows", "columns", "both"), ...)

### Arguments
- **df**: a 2 x k frequency table.
- **rev**: character string indicating whether to switch row or column order, possible options are "neither", "rows", "columns", or "both". The default is "neither".
- **...**: further arguments passed to or from other methods.

### Value
- **tab**: 2 x k frequency table
Examples

```r
trial <- data.frame(disease = rep("yes", 3), rep("no", 3)),
    treatment = rep(c("estrogen", "placebo", "other"), 2),
    count = c(751, 623, 7755, 7479, 9000, 456))

xtabs(count ~ treatment + disease, data = trial) %>% tavolo(rev = "columns")
```
Index

matched_or, 2
matched_or.data.frame, 3
matched_or.table, 4

prop_power, 5
prop_test, 6
prop_test.data.frame, 7
prop_test.matrix, 9
prop_test.numeric, 10
prop_test.table, 11

riskdiff, 12
riskdiff.data.frame, 13
riskdiff.matrix, 14
riskdiff.table, 15

tavolo, 16
tavolo.data.frame, 17
tavolo.matrix, 17
tavolo.table, 18