Package ‘permutest’

September 26, 2024

Title Run Permutation Tests and Construct Associated Confidence Intervals

Version 1.0.0

Description Implements permutation tests for any test statistic and randomization scheme and constructs associated confidence intervals as described in Glazer and Stark (2024) <doi:10.48550/arXiv.2405.05238>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Amanda Glazer [aut, cre] (<`https://orcid.org/0000-0002-3229-7924`>)

Maintainer Amanda Glazer <`amanda.glazer@austin.utexas.edu`>

Repository CRAN

Date/Publication 2024-09-26 11:20:02 UTC

Contents

`adjust_p_value` *Adjust p-values for multiple testing*

Description

This function takes an array of p-values and returns adjusted p-values using user-inputted FWER or FDR correction method

Usage

```
adjust_p_value(pvalues, method = "holm-bonferroni")
```

Arguments

`pvalues`	Array of p-values
`method`	The FWER or FDR correction to use, either 'holm-bonferroni', 'bonferroni', or 'benjamini-hochberg'

Value

Adjusted p-values

Examples

```
adjust_p_value(pvalues = c(.05, .1, .5), method='holm-bonferroni')
```

`diff_in_means` *Calculate difference in means*

Description

This function takes a data frame, and group and outcome column names as input and returns the difference in mean outcome between the two groups

Usage

```
diff_in_means(df, group_col, outcome_col, treatment_value = NULL)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`outcome_col`	The name of the column in df that corresponds to the outcome variable
`treatment_value`	The value of group_col to be considered 'treatment'

Value

The difference in mean outcome between the two groups

Examples

```
data <- data.frame(group = c(rep(1, 4), rep(2, 4)),
                   outcome = c(rep(3, 4), rep(5, 4)))

diff_in_means(df = data,
              group_col = "group",
              outcome_col = "outcome",
              treatment_value = 1)
```

`diff_in_medians` *Calculate difference in medians*

Description

This function takes a data frame, and group and outcome column names as input and returns the difference in median outcome between the two groups

Usage

```
diff_in_medians(df, group_col, outcome_col, treatment_value = NULL)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`outcome_col`	The name of the column in df that corresponds to the outcome variable
`treatment_value`	The value of group_col to be considered 'treatment'

Value

The difference in median outcome between the two groups

Examples

```
data <- data.frame(group = c(rep(1, 4), rep(2, 4)),
                   outcome = c(rep(3, 4), rep(5, 4)))

diff_in_medians(df = data,
              group_col = "group",
              outcome_col = "outcome",
              treatment_value = 1)
```

`fisher` *Fisher combining function*

Description

This function takes an array of p-values and returns a combined p-value using fisher's combining function: $-2\sum_i \log(p_i)$

Usage

```
fisher(pvalues)
```

Arguments

`pvalues`	Array of p-values

Value

Combined p-value using fisher's method

Examples

```
fisher(pvalues = c(.05, .1, .5))
```

`liptak` *Liptak combining function*

Description

This function takes an array of p-values and returns a combined p-value using Liptak's combining function: $\sum_i \Phi^{-1}(1 - p_i)$ where Φ is the CDF of the Normal distribution

Usage

```
liptak(pvalues)
```

Arguments

`pvalues`	Array of p-values

Value

Combined p-value using Liptak's method

Examples

```
liptak(pvalues = c(.05, .1, .5))
```

npc *Run NPC*

Description

This function takes a data frame and group and outcome column names as input and returns the nonparametric combination of tests (NPC) omnibus p-value

Usage

```
npc(
  df,
  group_col,
  outcome_cols,
  strata_col = NULL,
  test_stat = "diff_in_means",
  perm_func = permute_group,
  combn = "fisher",
  shift = 0,
  reps = 10000,
  perm_set = NULL,
  complete_enum = FALSE,
  seed = NULL
)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`outcome_cols`	The names of the columns in df that corresponds to the outcome variable
`strata_col`	The name of the column in df that corresponds to the strata
`test_stat`	Test statistic function
`perm_func`	Function to permute group, default is permute_group which randomly permutes group assignment

`combn`	Combining function method to use, takes values 'fisher', 'tippett', or 'liptak', or a user defined function
`shift`	Value of shift to apply in one- or two-sample problem
`reps`	Number of iterations to use when calculating permutation p-value
`perm_set`	Matrix of permutations to use instead of reps iterations of perm_func
`complete_enum`	Boolean, whether to calculate P-value under complete enumeration of permutations
`seed`	An integer seed value

Value

The omnibus p-value

Examples

```
data <- data.frame(group = c(rep(1, 4), rep(2, 4)),
out1 = c(0, 1, 0, 0, 1, 1, 1, 0),
out2 = rep(1, 8))
output <- npc(df = data, group_col = "group",
              outcome_cols = c("out1", "out2"), perm_func = permute_group,
              combn = "fisher", reps = 10^4, seed=42)
```

`one_sample` *One-sample permutation test*

Description

This function runs a permutation test for the one-sample problem by calling the permutation_test function using the one-sample mean test statistic.

Usage

```
one_sample(x, shift = 0, alternative = "greater", reps = 10^4, seed = NULL)
```

Arguments

`x`	array of data
`shift`	Value of shift to apply in one-sample problem
`alternative`	String, two-sided or one-sided (greater or less) p-value
`reps`	Number of iterations to use when calculating permutation p-value
`seed`	An integer seed value

Value

The permutation test p-value

Examples

```
one_sample(x = c(-1, 1, 2), seed = 42)
```

`one_sample_mean` *Calculate the one-sample problem test statistic*

Description

This function takes a data frame, and group and outcome column names as input and returns the mean of the product of the outcome and group. This test statistic is used for the one-sample problem.

Usage

```
one_sample_mean(df, group_col, outcome_col)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`outcome_col`	The name of the column in df that corresponds to the outcome variable

Value

The one-sample problem test statistic: the mean of the product of the outcome and group

Examples

```
data <- data.frame(group = c(rep(1, 4), rep(2, 4)),
                   outcome = c(rep(3, 4), rep(5, 4)))

one_sample_mean(df = data,
              group_col = "group",
              outcome_col = "outcome")
```

`one_way_anova_stat` *Calculate one-way anova test statistic*

Description

This function takes a data frame, and group and outcome column names as input and returns the one-way anova test statistic

Usage

```
one_way_anova_stat(df, group_col, outcome_col)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`outcome_col`	The name of the column in df that corresponds to the outcome variable

Value

The one-way anova test statistic: $\sum_{g=1}^{G} n_g(\overline{X_g} - \overline{X})^2$ where g indexes the groups

`permutation_test` *Run permutation test*

Description

Run permutation test with user inputted data, test statistic, and permutation function

Usage

```
permutation_test(
  df,
  group_col,
  outcome_col,
  strata_col = NULL,
  test_stat = "diff_in_means",
  perm_func = permute_group,
  alternative = "two-sided",
  shift = 0,
  reps = 10000,
  perm_set = NULL,
  complete_enum = FALSE,
  return_test_dist = FALSE,
  return_perm_dist = FALSE,
  seed = NULL
)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`outcome_col`	The name of the column in df that corresponds to the outcome variable
`strata_col`	The name of the column in df that corresponds to the strata
`test_stat`	Test statistic function
`perm_func`	Function to permute group
`alternative`	String, two-sided or one-sided (greater or less) p-value; options are 'greater', 'less', or 'two-sided'
`shift`	Value of shift to apply in one- or two-sample problem
`reps`	Number of iterations to use when calculating permutation p-value
`perm_set`	Matrix of group assignments to use instead of reps iterations of perm_func
`complete_enum`	Boolean, whether to calculate P-value under complete enumeration of permutations
`return_test_dist`	Boolean, whether to return test statistic distribution under permutations
`return_perm_dist`	Boolean, whether to return a matrix where each row is the group assignment under that permutation
`seed`	An integer seed value

Value

`p_value`: the permutation test p-value

`test_stat_dist`: array, the distribution of the test statistic under the set of permutations, if return_test_dist is set to TRUE

`perm_indices_mat`: matrix, each row corresponds to a permutation used in the permutation test calculation

Examples

```
data <- data.frame(group = c(rep(1, 10), rep(2, 10)), outcome = c(rep(1, 10), rep(1, 10)))

permutation_test(df = data, group_col = "group", outcome_col = "outcome",
test_stat = "diff_in_means", perm_func = permute_group, alternative = "greater",
shift = 0, reps = 10, return_perm_dist = TRUE, return_test_dist = TRUE, seed = 42)
```

permutation_test_ci *Construct confidence interval by inverting permutation tests*

Description

This function constructs a confidence interval by inverting permutation tests and applying the method in Glazer and Stark, 2024.

Usage

```
permutation_test_ci(
  df,
  group_col,
  outcome_col,
  strata_col = NULL,
  test_stat = "diff_in_means",
  perm_func = permute_group,
  upper_bracket = NULL,
  lower_bracket = NULL,
  cl = 0.95,
  e = 0.1,
  reps = 10000,
  perm_set = NULL,
  seed = 42
)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`outcome_col`	The name of the column in df that corresponds to the outcome variable
`strata_col`	The name of the column in df that corresponds to the strata
`test_stat`	Test statistic function
`perm_func`	Function to permute group
`upper_bracket`	Array with 2 values that bracket upper confidence bound
`lower_bracket`	Array with 2 values that bracket lower confidence bound
`cl`	Confidence level, default 0.95
`e`	Maximum distance from true confidence bound value
`reps`	Number of iterations to use when calculating permutation p-value
`perm_set`	Matrix of group assignments to use instead of reps iterations of perm_func
`seed`	An integer seed value

Value

A list containing the permutation test p-value, and the test statistic distribution if applicable

Examples

```
x <- c(35.3, 35.9, 37.2, 33.0, 31.9, 33.7, 36.0, 35.0, 33.3, 33.6, 37.9, 35.6, 29.0, 33.7, 35.7)
y <- c(32.5, 34.0, 34.4, 31.8, 35.0, 34.6, 33.5, 33.6, 31.5, 33.8, 34.6)
df <- data.frame(outcome = c(x, y), group = c(rep(1, length(x)), rep(0, length(y))))
permutation_test_ci(df = df, group_col = "group", outcome_col = "outcome", strata_col = NULL,
                    test_stat = "diff_in_means", perm_func = permute_group,
                    upper_bracket = NULL, lower_bracket = NULL,
                    cl = 0.95, e = 0.01, reps = 10^3, seed = 42)
```

`permute_group` *Unstratified group permutation*

Description

This function takes a data frame and group column name as input and returns the dataframe with the group column randomly permuted

Usage

```
permute_group(df, group_col, strata_col = NULL, seed = NULL)
```

Arguments

`df`	A data frame
`group_col`	String, the name of the column in df that corresponds to the group label
`strata_col`	The name of the column in df that corresponds to the strata, should be NULL for unstratified permutation
`seed`	An integer seed value

Value

The inputted data frame with the group column randomly shuffled

Examples

```
data <- data.frame(group_label = c(1, 2, 2, 1, 2, 1), outcome = 1:6)
permute_group(df = data, group_col = "group_label", strata_col = NULL, seed = 42)
```

`permute_sign` *Sign permutation*

Description

This function takes a data frame and group and outcome column name as input and returns the dataframe with the group column replaced with randomly assigned signs

Usage

```
permute_sign(df, group_col, strata_col = NULL, seed = NULL)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`strata_col`	The name of the column in df that corresponds to the strata, should be NULL for this function
`seed`	An integer seed value

Value

The inputted data frame with the group column replaced with randomly assigned signs

Examples

```
data <- data.frame(group_label = rep(1, 6), outcome = 1:6)
permute_group(df = data, group_col = "group_label", strata_col = NULL, seed = 42)
```

`strat_permute_group` *Stratified group permutation*

Description

This function takes a data frame and group and strata column name as input and returns the dataframe with the group column randomly permuted by strata

Usage

```
strat_permute_group(df, group_col, strata_col, seed = NULL)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`strata_col`	The name of the column in df that corresponds to the strata
`seed`	An integer seed value

Value

The inputted data frame with the group column randomly shuffled by strata

Examples

```
data <- data.frame(group_label = c(1, 2, 2, 1, 2, 1), stratum = c(1, 1, 1, 2, 2, 2), outcome = 1:6)
permute_group(df = data, group_col = "group_label", strata_col = "stratum", seed = 42)
```

`tippett` *Tippett combining function*

Description

This function takes an array of p-values and returns a combined p-value using Tippett's combining function: $\max_i\{1 - p_i\}$

Usage

```
tippett(pvalues)
```

Arguments

`pvalues`	Array of p-values

Value

Combined p-value using Tippett's method

Examples

```
tippett(pvalues = c(.05, .1, .5))
```

`ttest_stat` *Calculate t-test statistic*

Description

This function takes a data frame, and group and outcome column names as input and returns the t test statistic

Usage

```
ttest_stat(df, group_col, outcome_col)
```

Arguments

`df`	A data frame
`group_col`	The name of the column in df that corresponds to the group label
`outcome_col`	The name of the column in df that corresponds to the outcome variable

Value

The t test statistic

`two_sample` *Two-sample permutation test*

Description

This function runs a permutation test with difference in means test statistic for the two-sample problem by calling the permutation_test function.

Usage

```
two_sample(x, y, shift = 0, alternative = "greater", reps = 10^4, seed = NULL)
```

Arguments

`x`	array of data for treatment group
`y`	array of data for control group
`shift`	Value of shift to apply in two-sample problem
`alternative`	String, two-sided or one-sided (greater or less) p-value; options are ’greater’, ’less’, or ’two-sided’
`reps`	Number of iterations to use when calculating permutation p-value
`seed`	An integer seed value

Value

The permutation test p-value

Examples

```
two_sample(x = c(10, 9, 11), y = c(12, 11, 13), alternative = "less", seed = 42)
```

Index