Package 'mmibain'

November 8, 2023

Title Bayesian Informative Hypotheses Evaluation Web Applications

Version 0.1.1

Description Researchers often have expectations about the relations between means of different groups or standardized regression coefficients; using informative hypothesis testing to incorporate these expectations into the analysis through order constraints increases statistical power Vanbrabant and Rosseel (2020) <doi:10.4324/9780429273872-14>. Another valuable tool, the Bayes factor, can evaluate evidence for multiple hypotheses without concerns about multiple testing, and can be used in Bayesian updating Hoijtink, Mulder, van Lissa & Gu (2019) <doi:10.1037/met0000201>. The 'bain' R package enables informative hypothesis testing using the Bayes factor. The 'mmibain' package provides 'shiny' web applications based on 'bain'. The RepliCrisis() function launches a 'shiny' card game to simulate the evaluation of replication studies while the mmibain() function launches a 'shiny' application to fit Bayesian informative hypotheses evaluation models from 'bain'.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

URL https://github.com/mightymetrika/mmibain

BugReports https://github.com/mightymetrika/mmibain/issues

Imports bain, broom, car, e1071, igraph, lavaan, mmcards, psych, shiny, shinythemes

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Mackson Ncube [aut, cre], mightymetrika, LLC [cph, fnd]

Maintainer Mackson Ncube <macksonncube.stats@gmail.com>

Repository CRAN

Date/Publication 2023-11-08 20:00:02 UTC

R topics documented:

`deal_cards_to_rc_grid` *Deal Cards to a RepliCrisis Grid*

Description

This function deals cards from a shuffled deck and arranges them into a matrix suitable for the RepliCrisis game grid.

Usage

```
deal_cards_to_rc_grid(
  deck = mmcards::i_deck(deck = mmcards::shuffle_deck(), i_path = "www", i_names =
    c("2_of_clubs", "2_of_diamonds", "2_of_hearts", "2_of_spades", "3_of_clubs",
   "3_of_diamonds", "3_of_hearts", "3_of_spades", "4_of_clubs", "4_of_diamonds",
    "4_of_hearts", "4_of_spades", "5_of_clubs", "5_of_diamonds", "5_of_hearts",
    "5_of_spades", "6_of_clubs", "6_of_diamonds", "6_of_hearts", "6_of_spades",
    "7_of_clubs", "7_of_diamonds", "7_of_hearts", "7_of_spades", "8_of_clubs",
    "8_of_diamonds", "8_of_hearts", "8_of_spades",
     "9_of_clubs", "9_of_diamonds",
   "9_of_hearts", "9_of_spades", "10_of_clubs", "10_of_diamonds", "10_of_hearts",
    "10_of_spades", "jack_of_clubs", "jack_of_diamonds", "jack_of_hearts",
    "jack_of_spades", "queen_of_clubs", "queen_of_diamonds", "queen_of_hearts",
    "queen_of_spades", "king_of_clubs", "king_of_diamonds", "king_of_hearts",
    "king_of_spades", "ace_of_clubs", "ace_of_diamonds", "ace_of_hearts",
    "ace_of_spades")),
  n
)
```

Arguments

`deck`	A data frame representing a deck of cards, which by default is a shuffled standard deck from the `mmcards` package.
`n`	The number of card pairs to deal. The function will deal 2*n cards and arrange them into two rows, for the RepliCrisis grid.

Details

The function first checks if there are enough cards in the deck to deal the required number of pairs. If not, it stops with an error. Then, it deals 2*n cards from the provided deck, reshaping them into a 2-row matrix where each column represents a pair of cards.

If no deck is provided, the function will shuffle a standard deck using functions from the `mmcards` package. The default deck includes all standard 52 playing cards.

The grid of cards will be used by the generate_study_data function to generate data for n groups where the values for each group are simulated from a normal distribution with mean and standard deviation defined by the values in the card pair.

Value

A matrix with two rows and n columns, representing the dealt cards arranged into pairs.

Examples

```
# Deal a grid with 3 card pairs
grid <- deal_cards_to_rc_grid(n = 3)
```

`generate_Ho_from_pairwise_t` *Generate Null Hypothesis from Pairwise t-test Results*

Description

This function generates a null hypothesis statement (Ho) from the results of pairwise t-tests. If pairwise t-tests are not available, it uses the hypothesis from the original study results.

Usage

```
generate_Ho_from_pairwise_t(original_study_results)
```

Arguments

`original_study_results`
: A list containing the results of an original study, including a `pairwise_t` element if pairwise t-tests were conducted.

Details

The function checks if the `pairwise_t` element is present in the `original_study_results` list. If present, it extracts the p-values and the corresponding column and row names to identify all unique variables involved in the pairwise comparisons.

The unique variables are prefixed with "ColLabs" to denote the column labels from the original dataset. These variables are then concatenated with an equality sign to form the null hypothesis statement, which assumes no difference between any of the groups.

If the `pairwise_t` element is not present, indicating that no pairwise comparisons were made, the function returns the hypothesis generated by the original study results.

Value

A string representing the null hypothesis for the study.

Examples

```
Ho <- generate_Ho_from_pairwise_t(
          original_study_results = process_original_study(
                                   df = generate_study_data(
                                   x = deal_cards_to_rc_grid(n = 3),
                                   sample_size = 30),
                                   alpha = 0.05))
```

---

generate_study_data *Generate Study Data for RepliCrisis*

Description

This function simulates data for the Replication Crisis study by drawing samples from normal distributions defined by the card values.

Usage

```
generate_study_data(x, sample_size)
```

Arguments

x	A matrix with two rows representing the mean and standard deviation for each group.
sample_size	The number of samples to draw for each study group.

Details

The function expects a matrix x generated from deal_cards_to_rc_grid() where the first row contains mean values and the second row contains standard deviation values. It then generates sample_size number of normal random values for each group, using the respective mean and standard deviation. The resulting data frame has two columns: one for the group labels and one for the generated values.

The group labels are factors with levels corresponding to the column numbers prefixed by 'Col'. The generated values are numeric and simulate the data that would be collected in the study. The function uses the rnorm function from the stats package for generating random samples.

Value

A data frame containing the simulated study data. Each row corresponds to a single sample and includes the group label and the sampled value.

Examples

```
study_data <- generate_study_data(x = deal_cards_to_rc_grid(n = 3),
                                  sample_size = 30)
```

`interpret_replication_results` *Interpretation of Replication Study Results*

Description

This function interprets the results of a replication study by applying Bayes factor (BF) and posterior model probabilities (PMPb) thresholds. It checks against predefined thresholds to determine if there is strong evidence for the hypotheses derived from the original study.

Usage

```
interpret_replication_results(
  replication_results,
  bf_threshold = 3,
  pmpb_threshold = 0.8
)
```

Arguments

`replication_results`	A list containing the results of a replication study, specifically a bain object with BF and PMPb values.
`bf_threshold`	The threshold for the Bayes factor (BF.c) above which the evidence is considered strong. Default is 3.
`pmpb_threshold`	The threshold for the posterior model probabilities (PMPb) above which the evidence is considered strong. Default is 0.80.

Details

The function first checks for the presence of a secondary hypothesis (H2) in the analysis results. If H2 is present, it will prioritize its interpretation; otherwise, it defaults to interpreting H1. Interpretation is based on whether the BF.c and PMPb values exceed their respective thresholds.

A 'win' result means there is strong evidence for the hypothesis, while a 'lose' indicates the evidence is inconclusive or not strong enough.

The function includes a disclaimer about the use of threshold values for hypothesis testing and recommends consulting the cited literature for a comprehensive understanding of Bayesian factors and informative hypothesis testing.

Value

A list with elements 'interpretation' providing the interpretative message, 'result' indicating a 'win' or 'lose' based on the interpretation, and 'disclaimer' providing a contextual disclaimer.

Note

The thresholds used in this function are for educational purposes within the context of a game and should not be taken as rigid rules for hypothesis testing in practice.

Examples

```
# Original study
os_deck <- deal_cards_to_rc_grid(n = 3)
original_study_data <- generate_study_data(os_deck, sample_size = 100)
original_study_results <- process_original_study(original_study_data)

# Replication study
rs_deck <- deal_cards_to_rc_grid(n = 3)
replication_data <- generate_study_data(rs_deck, sample_size = 100)
replication_results <- process_replication_study(replication_data,
                                                 original_study_results)

interpret_replication_results(replication_results)
```

---

mmibain *mmibain Shiny App*

Description

The user can upload CSV data; choose a model engine (lm, t_test, lavaan); specify the formula, variables, or model; and provide additional arguments. Once the model is fitted, the app allows for setting up hypotheses for evaluation. Upon running the analysis, it displays the results of the Bayesian Informative Hypotheses Evaluation.

Usage

```
mmibain()
```

Details

This function launches a Shiny app that facilitates a user-friendly interface for setting up and running a Bayesian Informative Hypotheses Evaluation using the `bain` package.

The app's UI consists of a sidebar for user inputs and a main panel for displaying available variables, model terms, and analysis results. The app relies on the `bain` package for analysis.

Value

This function returns a running instance of the Shiny app. Interact with the app through the browser or the RStudio Viewer pane.

UI Components

- Data upload (CSV format).
- Engine selection (lm, t_test, lavaan).
- Model input based on chosen engine.
- Additional arguments for statistical model function.
- Action button to fit the model.
- Hypotheses input.
- Fraction input for the `bain` fraction parameter.
- Option to evaluate hypotheses with respect to standardized regression coefficients.
- Confidence interval input.
- Seed input for reproducibility.
- Action button to run the Bayesian Informative Hypotheses Evaluation.

References

Hoijtink, H., Mulder, J., van Lissa, C., & Gu, X. (2019). A tutorial on testing hypotheses using the Bayes factor. Psychological methods, 24(5), 539–556. https://doi.org/10.1037/met0000201

See Also

`bain`

Examples

```
if(interactive()){
  mmibain()
}
```

---

mmib_model *Fit Statistical Models for MMI Bain Processing*

Description

This function provides a unified interface to fit different statistical models supported by the 'bain' package.

Usage

```
mmib_model(
  formula = NULL,
  column_names = NULL,
  model = NULL,
  data,
  engine = c("lm", "t_test", "lavaan"),
  ...
)
```

Arguments

formula	A symbolic description of the model to be fit. Used specifically for the lm engine. Default is NULL.
column_names	A character vector of length 2, representing the column names to be used for the t_test engine. Default is NULL.
model	A model specification (usually as a string) for the lavaan engine. Default is NULL.
data	A data frame containing the variables in the model.
engine	A character string representing the statistical method to be used. Currently supported methods are: "lm", "t_test", and "lavaan".
...	Additional arguments to be passed to the underlying statistical function (lm(), t.test(), or lavaan::sem()).

Details

The mmib_model() function provides a simple interface to fit various statistical models, which can be subsequently processed by the bain::bain() function. It ensures that only one of formula, column_names, or model is provided, checks the validity of the provided data, and selects the appropriate statistical method based on the engine parameter.

Value

Returns an object of the type associated with the engine selected (lm, htest, or lavaan object).

See Also

bain for processing the models fit with mmib_model.

Examples

```
data(mtcars)

# Fit linear model
mod1 <- mmib_model(mpg ~ wt + qsec, data = mtcars, engine = "lm")
```

---

process_original_study

Process Original Study Data for Analysis

Description

This function processes the original study data by performing ANOVA, post-hoc t-tests, and checking assumptions such as normality of residuals and homogeneity of variances.

Usage

```
process_original_study(df, alpha = 0.05)
```

Arguments

df	A data frame containing the study data, with columns ColLabs for labels and ColVals for values.
alpha	Significance level for the statistical tests, by default set to 0.05.

Details

The function starts by fitting an ANOVA model to the data to check for overall significance. If significant differences are found, it proceeds with pairwise t-tests to explore differences between individual conditions.

The function then checks for normality of residuals using the Shapiro-Wilk test and for homogeneity of variances using Levene's test. It constructs a hypothesis string based on the results of the pairwise t-tests.

A directed graph is created to represent the relationships between the conditions. The graph is then simplified to reflect the most direct relationships, which are used to construct the final hypothesis string.

Descriptive statistics are generated for each condition using the generate_descriptives() internal function.

The function returns a list with the following components:

- hypothesis: A string representing the simplified relationships between conditions.
- pairwise_t: The result of pairwise t-tests, if performed.
- fit_summary: The summary of the ANOVA model.
- descriptives: Descriptive statistics for each condition.
- fit: The ANOVA model object.
- shapiro_test: The result of the Shapiro-Wilk normality test.
- levene_test: The result of Levene's test for homogeneity of variances.

Value

A list containing elements for hypothesis, pairwise t-tests (if applicable), fit summary, descriptives, fit object, Shapiro-Wilk normality test result, and Levene's test result.

Examples

```
results <- process_original_study(df = generate_study_data(
                                      x = deal_cards_to_rc_grid(n = 3),
                                      sample_size = 30),
                                  alpha = 0.05)
```

process_replication_study *Process Replication Study Data*

Description

This function processes the data from a replication study by comparing it with the results from an original study, using the bain package for Bayesian inference.

Usage

```
process_replication_study(replication_data, original_study_results)
```

Arguments

replication_data	A data frame containing the replication study data, with columns ColLabs for labels and ColVals for values.
original_study_results	A list containing the results of an original study, including the hypothesis and pairwise t-test results, if applicable.

Details

The function begins by extracting the original hypothesis (Horiginal) from the original_study_results. It then generates the null hypothesis (Ho) using the generate_Ho_from_pairwise_t function.

A hypothesis string for the Bayesian analysis is prepared by comparing Ho and Horiginal. If they are the same, only Horiginal is used; otherwise, both are included in the Bayesian analysis via bain.

An ANOVA model is fitted to the replication data, which is then passed to the bain function along with the hypothesis string. The Bayesian analysis results are returned along with any message regarding the comparison of hypotheses.

Value

A list containing the results from the Bayesian analysis (bain_results) and a message indicating if the null hypothesis and the original hypothesis were the same (message).

Examples

```
replication_results <- process_replication_study(
  replication_data = generate_study_data(
    x = deal_cards_to_rc_grid(n = 3),
    sample_size = 30
  ),
  original_study_results = process_original_study(
    df = generate_study_data(
      x = deal_cards_to_rc_grid(n = 3),
```

```
      sample_size = 30
    ),
    alpha = 0.05
  )
)
```

`RepliCrisis` *RepliCrisis Shiny App*

Description

The RepliCrisis Shiny app is an interactive game based on the process for evaluating replication studies presented in Hoijtink et. al. (2009). Users can conduct original studies, process the results, and attempt to replicate the studies while adjusting parameters to see how changes affect the outcome.

Usage

```
RepliCrisis()
```

Value

An interactive Shiny app.

References

Hoijtink, H., Mulder, J., van Lissa, C., & Gu, X. (2019). A tutorial on testing hypotheses using the Bayes factor. Psychological methods, 24(5), 539–556. https://doi.org/10.1037/met0000201

Examples

```
if(interactive()){
RepliCrisis()
}
```

`swapper` *Card Matrix Swapping Function*

Description

This function performs swapping operations within a card matrix to rearrange cards. It can swap entire columns, swap cards within a single column, or swap cards within a single row. It is designed to be used on a replication study card grid before generating replication study data.

Usage

```
swapper(cards_matrix, swap_cols = NULL, swap_in_col = NULL, swap_in_row = NULL)
```

Arguments

cards_matrix	A matrix representing the card grid on which to perform swaps.
swap_cols	A numeric vector of length 2 specifying the columns to swap.
swap_in_col	A single integer indicating a column where the two cards will be swapped.
swap_in_row	A numeric vector of length 3 indicating the row number followed by the two column numbers within that row to swap.

Details

The swapper function can be used to rearrange cards in a card matrix. It allows for three types of swaps:

- Swapping two columns: Specify two columns to swap their entire content.
- Swapping within a column: Reverse the order of two cards in the same column.
- Swapping within a row: Swap two cards within the same row.

The function keeps track of the swapping history to prevent multiple swaps within the same column or row, as these are restricted operations. After a swap operation, the function updates the class of the cards_matrix to include "swapper" to track its history.

Value

A matrix of the same dimensions as cards_matrix with the specified swaps performed.

Examples

```
swapper(cards_matrix = deal_cards_to_rc_grid(n = 3), swap_cols = c(1,2))
```

Index