Package 'infer'

January 22, 2018

Type Package

Title Tidy Statistical Inference

Version 0.1.1

Description The objective of this package is to perform inference using an expressive statistical grammar that coheres with the tidy design framework.

License CC0

Encoding UTF-8

LazyData true

Imports assertive, dplyr (>= 0.7.0), methods, tibble, rlang, ggplot2, magrittr

Depends R (>= 3.1.2)

Suggests broom, devtools (>= 1.12.0), knitr, rmarkdown, nycflights13, stringr, testthat, covr

URL https://github.com/andrewpbray/infer

BugReports https://github.com/andrewpbray/infer/issues

RoxygenNote 6.0.1

VignetteBuilder knitr

NeedsCompilation no

Author Andrew Bray [aut, cre],
Chester Ismay [aut],
Ben Baumer [aut],
Mine Cetinkaya-Rundel [aut],
Ted Laderas [ctb],
Nick Solomon [ctb],
Johanna Hardin [ctb],
Albert Kim [ctb]

Maintainer Andrew Bray <abray@reed.edu>

Repository CRAN

Date/Publication 2018-01-22 06:59:42 UTC

R topics documented:

`calculate` *Calculate summary statistics*

Description

Calculate summary statistics

Usage

```
calculate(x, stat, order = NULL, ...)
```

Arguments

`x`	the output from `generate` for computation-based inference. Down the road, `hypothesize` will also be able to be piped in to here for theory-based inference.
`stat`	a string giving the type of the statistic to calculate. Current options include "mean", "median", "sd", "prop", "diff in means", "diff in medians", "diff in props", "Chisq", "F", and "slope".
`order`	a string vector of specifying the order in which the levels of the explanatory variable should be ordered for subtraction, where `order = c("first", "second")` means `("first" - "second")` Needed for inference on difference in means, medians, or proportions.
`...`	to pass options like `na.rm = TRUE` into functions like mean, sd, etc.

Value

A tibble containing a `stat` column of calculated statistics

Examples

```
# Permutation test for two binary variables
  mtcars %>%
    dplyr::mutate(am = factor(am), vs = factor(vs)) %>%
    specify(am ~ vs, success = "1") %>%
    hypothesize(null = "independence") %>%
    generate(reps = 100, type = "permute") %>%
    calculate(stat = "diff in props", order = c("1", "0"))
```

`generate` *Generate resamples, permutations, or simulations based on 'specify' and 'hypothesize' inputs*

Description

Generate resamples, permutations, or simulations based on 'specify' and 'hypothesize' inputs

Usage

```
generate(x, reps = 1, type = "bootstrap", ...)
```

Arguments

`x`	a data frame that can be coerced into a `tbl_df`
`reps`	the number of resamples to generate
`type`	currently either `bootstrap`, `permute`, or `simulate`
`...`	currently ignored

Value

A tibble containing `rep` generated datasets, indicated by the `replicate` column.

Examples

```
# Permutation test for two binary variables
  mtcars %>%
    dplyr::mutate(am = factor(am), vs = factor(vs)) %>%
    specify(am ~ vs, success = "1") %>%
    hypothesize(null = "independence") %>%
    generate(reps = 100, type = "permute")
```

hypothesize *Declare a null hypothesis*

Description

Declare a null hypothesis

Usage

```
hypothesize(x, null, ...)
```

Arguments

x	a data frame that can be coerced into a tbl_df
null	the null hypothesis. Options include "independence" and "point"
...	arguments passed to downstream functions

Value

A tibble containing the response (and explanatory, if specified) variable data with parameter information stored as well

Examples

```
# Permutation test similar to ANOVA
  mtcars %>%
    dplyr::mutate(cyl = factor(cyl)) %>%
    specify(mpg ~ cyl) %>%
    hypothesize(null = "independence") %>%
    generate(reps = 100, type = "permute") %>%
    calculate(stat = "F")
```

infer *infer: a grammar for statistical inference*

Description

The objective of this package is to perform statistical inference using a grammar that illustrates the underlying concepts and a format that coheres with the tidyverse.

Examples

```
# Example usage:
library(infer)
```

print.infer *Print methods*

Description

Print methods

Usage

```
## S3 method for class 'infer'
print(x, ...)
```

Arguments

x	an object of class infer, i.e. output from specify or hypothesize
...	arguments passed to methods

rep_sample_n *Perform repeated sampling*

Description

Perform repeated sampling of samples of size n. Useful for creating sampling distributions

Usage

```
rep_sample_n(tbl, size, replace = FALSE, reps = 1, prob = NULL)
```

Arguments

tbl	data frame of population from which to sample
size	sample size of each sample
replace	should sampling be with replacement?
reps	number of samples of size n = size to take
prob	a vector of probability weights for obtaining the elements of the vector being sampled.

Value

A tibble of size rep times size rows corresponding to rep samples of size n = size from tbl.

Examples

```
suppressPackageStartupMessages(library(dplyr))
suppressPackageStartupMessages(library(ggplot2))

# Create a virtual population of N = 2400 balls, of which 900 are red and the
# rest are white
N <- 2400
population <- data_frame(
  ball_ID = 1:N,
  color = c(rep("red", 900), rep("white", N - 900))
)
population

# Take samples of size n = 50 balls without replacement; do this 1000 times
samples <- population %>%
  rep_sample_n(size = 50, reps = 1000)
samples

# Compute p_hats for all 1000 samples = proportion red
p_hats <- samples %>%
  group_by(replicate) %>%
  summarize(prop_red = mean(color == "red"))
p_hats

# Plot sampling distribution
ggplot(p_hats, aes(x = prop_red)) +
  geom_density() +
  labs(x = "p_hat", y = "Number of samples",
  title = "Sampling distribution of p_hat from 1000 samples of size 50")
```

`specify` *Specify the response and explanatory variables*

Description

Specify the response and explanatory variables

Usage

```
specify(x, formula, response = NULL, explanatory = NULL, success = NULL)
```

Arguments

`x`	a data frame that can be coerced into a `tbl_df`
`formula`	a formula with the response variable on the left and the explanatory on the right
`response`	the variable name in `x` that will serve as the response. This is alternative to using the `formula` argument
`explanatory`	the variable name in `x` that will serve as the explanatory variable

success	the level of response that will be considered a success, as a string. Needed for inference on one proportion or a difference in proportions

Value

A tibble containing the response (and explanatory, if specified) variable data

Examples

```
# Permutation test similar to ANOVA
  mtcars %>%
    dplyr::mutate(cyl = factor(cyl)) %>%
    specify(mpg ~ cyl) %>%
    hypothesize(null = "independence") %>%
    generate(reps = 100, type = "permute") %>%
    calculate(stat = "F")
```

---

visualize *(Currently) Visualize the randomization-based distribution (To be updated to include theory-based distributions)*

Description

(Currently) Visualize the randomization-based distribution (To be updated to include theory-based distributions)

Usage

```
visualize(df, bins = 30, ...)
```

Arguments

df	the output from calculate
bins	the number of bins in the histogram
...	currently ignored

Value

A ggplot object showing the randomization-based distribution as a histogram. Preferable to use the ggplot2 package directly here as a histogram does not always display the distribution well.

`%>%` *Pipe*

Description

Like dplyr, infer also uses the pipe function, `%>%` to turn function composition into a series of imperative statements.

Arguments

`lhs, rhs`	Inference functions

Index