Package ‘moderndive’

November 4, 2019

Type Package

Title Tidyverse-Friendly Introductory Linear Regression

Version 0.4.0

Maintainer Albert Y. Kim <albert.ys.kim@gmail.com>


Depends R (>= 3.4.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/ModernDive/moderndive_package

BugReports https://github.com/ModernDive/moderndive_package/issues

Imports magrittr, dplyr, ggplot2, tibble, janitor, broom (>= 0.4.3), formula.tools, stringr, knitr, infer, rlang (>= 0.2.0), glue

RoxygenNote 6.1.1

Suggests testthat, covr, rmarkdown, vdiffr

VignetteBuilder knitr

NeedsCompilation no

Author Albert Y. Kim [aut, cre], Chester Ismay [aut], Andrew Bray [ctb], Delaney Moran [ctb], Evgeni Chasnovski [ctb]

Repository CRAN

Date/Publication 2019-11-04 05:40:02 UTC
R topics documented:

- bowl
- bowl_samples
- bowl_sample_1
- DD_vs_SB
- evals
- geom_parallel_slopes
- get_correlation
- get_regression_points
- get_regression_summaries
- get_regression_table
- house_prices
- MA_schools
- moderndive
- movies_sample
- mythbusters_yawn
- orig_pennies_sample
- pennies
- pennies_resamples
- pennies_sample
- promotions
- promotions_shuffled
- tactile_prop_red

Index

bowl A sampling bowl of red and white balls

Description

A sampling bowl used as the population in a simulated sampling exercise. Also known as the urn sampling framework https://en.wikipedia.org/wiki/Urn_problem.

Usage

bowl

Format

A data frame 2400 rows representing different balls in the bowl, of which 900 are red and 1500 are white.

ball_ID ID variable used to denote all balls. Note this value is not marked on the balls themselves

color color of ball: red or white
Examples

library(dplyr)
library(ggplot2)

# Take 10 different samples of size n = 50 balls from bowl
bowl_samples_simulated <- bowl %>%
  rep_sample_n(50, reps = 10)

# Compute 10 different p_hats (prop red) based on 10 different samples of
# size n = 50
p_hats <- bowl_samples_simulated %>%
  group_by(replicate, color) %>%
  summarize(count = n()) %>%
  mutate(proportion = count/50) %>%
  filter(color == "red")

# Plot sampling distribution
ggplot(p_hats, aes(x = proportion)) +
  geom_histogram(binwidth = 0.05) +
  labs(x = expression(hat(p)), y = "Number of samples",
       title = "Sampling distribution of p_hat based 10 samples of size n = 50")

bowl_samples

Sampling from a bowl of balls

Description

Counting the number of red balls in 10 samples of size n = 50 balls from https://github.com/moderndive/moderndive/blob/master/data-raw/sampling_bowl.jpeg

Usage

bowl_samples

Format

A data frame 10 rows representing different groups of students’ samples of size n = 50 and 5 variables

  group  Group name
  red    Number of red balls sampled
  white  Number of white balls sampled
  green  Number of green balls sampled
  n      Total number of balls samples

See Also

bowl
Examples

```r
library(dplyr)
library(ggplot2)

# Compute proportion red
bowl_samples <- bowl_samples %>%
  mutate(prop_red = red / n)

# Plot sampling distributions
ggplot(bowl_samples, aes(x = prop_red)) +
  geom_histogram(binwidth = 0.05) +
  labs(x = expression(hat(p)), y = "Number of samples",
       title = "Sampling distribution of p_hat based 10 samples of size n = 50")
```

---

bowl_sample_1  
Tactile sample of size 50 from a bowl of balls

Description


Usage

bowl_sample_1

Format

A data frame of 50 rows representing different balls and 1 variable.

- **color**  Color of ball sampled

See Also

bowl

Examples

```r
library(ggplot2)

# Barplot of distribution of balls in sample
ggplot(bowl_sample_1, aes(x = color)) +
  geom_bar() +
  labs(title = "50 sampled bals from bowl")
```
**Description**

Number of Dunkin Donuts & Starbucks, median income, and population in 1024 census tracts in eastern Massachusetts in 2016.

**Usage**

DD_vs_SB

**Format**

A data frame of 1024 rows representing census tracts and 6 variables

**county**  County where census tract is located. Either Bristol, Essex, Middlesex, Norfolk, Plymouth, or Suffolk county

**FIPS**  Federal Information Processing Standards code identifying census tract

**median_income**  Median income of census tract

**population**  Population of census tract

**shop_type**  Coffee shop type: Dunkin Donuts or Starbucks

**shops**  Number of shops

**Source**

US Census Bureau. Code used to scrape data available at https://github.com/DelaneyMoran/FinalProject

**Examples**

```r
# Compute correlation between a census tract's median income and number of cafes of
# each type after removing two cases where median_income is missing
library(dplyr)
DD_vs_SB %>%
  mutate(shops_per_1000 = 1000 * shops/population) %>%
  filter(!is.na(median_income)) %>%
  group_by(shop_type) %>%
  summarize(cor = cor(median_income, shops_per_1000))
```
Description
The data are gathered from end of semester student evaluations for a sample of 463 courses taught by 94 professors from the University of Texas at Austin. In addition, six students rate the professors’ physical appearance. The result is a data frame where each row contains a different course and each column has information on either the course or the professor https://www.openintro.org/stat/data/?data=evals

Usage
evals

Format
A data frame with 463 observations corresponding to courses on the following 13 variables.

ID Identification variable for course.
prof_ID Identification variable for professor. Many professors are included more than once in this dataset.
score Average professor evaluation score: (1) very unsatisfactory - (5) excellent.
age Age of professor.
bty_avg Average beauty rating of professor.
gender Gender of professor: female, male.
etnicity Ethnicity of professor: not minority, minority.
language Language of school where professor received education: English or non-English.
rank Rank of professor: teaching, tenure track, tenured.
pic_outfit Outfit of professor in picture: not formal, formal.
pic_color Color of professor’s picture: color, black & white.
cls_did_eval Number of students in class who completed evaluation.
cls_students Total number of students in class.
cls_level Class level: lower, upper.

Source

Examples
library(dplyr)
glimpse(evals)
Description

`geom_parallel_slopes()` fits parallel slopes model and adds its line output(s) to a `ggplot` object. Basically, it fits a unified model with intercepts varying between groups (which should be supplied as standard `ggplot2` grouping aesthetics: `group`, `color`, `fill`, etc.). This function has the same nature as `geom_smooth()` from `ggplot2` package, but provides functionality that `geom_smooth()` currently doesn’t have.

Usage

```
geom_parallel_slopes(mapping = NULL, data = NULL, position = "identity", ..., se = TRUE, formula = y ~ x, n = 100, na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)
```

Arguments

- **mapping**: Set of aesthetic mappings created by `aes()` or `aes()`. If specified and `inherit.aes` = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

- **data**: The data to be displayed in this layer. There are three options:
  - If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.
  - A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.
  - A function will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(.x, 10)`).

- **position**: Position adjustment, either as a string, or the result of a call to a position adjustment function.

- **...**: Other arguments passed on to `layer()`. These are often aesthetics, used to set an aesthetic to a fixed value, like `colour = "red"` or `size = 3`. They may also be parameters to the paired geom/stat.

- **se**: Display confidence interval around model lines? TRUE by default.

- **formula**: Formula to use per group in parallel slopes model. Basic linear `y ~ x` by default.

- **n**: Number of points per group at which to evaluate model.

- **na.rm**: If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.

- **show.legend**: logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.
If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behaviour from the default plot specification, e.g. `borders()`.

### Examples

```r
library(dplyr)
library(ggplot2)

ggplot(evals, aes(x = age, y = score, color = ethnicity)) +
  geom_point() +
  geom_parallel_slopes(se = FALSE)

# Basic usage
ggplot(evals, aes(x = age, y = score, color = ethnicity)) +
  geom_point() +
  geom_parallel_slopes()

# Supply custom aesthetics
ggplot(evals, aes(x = age, y = score, color = ethnicity)) +
  geom_point() +
  geom_parallel_slopes(se = FALSE, size = 4)

# Fit non-linear model
example_df <- house_prices %>%
  slice(1:1000) %>%
  mutate(
    log10_price = log10(price),
    log10_size = log10(sqft_living)
  )

ggplot(example_df, aes(x = log10_size, y = log10_price, color = condition)) +
  geom_point(alpha = 0.1) +
  geom_parallel_slopes(formula = y ~ poly(x, 2))

# Different grouping
ggplot(example_df, aes(x = log10_size, y = log10_price)) +
  geom_point(alpha = 0.1) +
  geom_parallel_slopes(aes(fill = condition))
```

---

### Description

Determine the Pearson correlation coefficient between two variables in a data frame using pipeable and formula-friendly syntax.
get_regression_points

Usage

```r
generate_regression_points(model, digits = 3, print = FALSE, newdata = NULL, ID = NULL)
```

Description

Output information on each point/observation used in an lm() regression in "tidy" format. This function is a wrapper function for broom::augment() and renames the variables to have more intuitive names.

Arguments

- `data`: a data frame object
- `formula`: a formula with the response variable name on the left and the explanatory variable name on the right
- `na.rm`: a logical value indicating whether NA values should be stripped before the computation proceeds.
- `...`: further arguments passed to `cor`

Value

A 1x1 data frame storing the correlation value

Examples

```r
library(moderndive)

# Compute correlation between mpg and cyl:
mtcars %>%
  generate_regression_points(formula = mpg ~ cyl)

# Group by one variable:
library(dplyr)
mtcars %>%
  group_by(am) %>%
  generate_regression_points(formula = mpg ~ cyl)

# Group by two variables:
mtcars %>%
  group_by(am, gear) %>%
  generate_regression_points(formula = mpg ~ cyl)
```
get_regression_points

Arguments

model  an \texttt{lm()} model object

digits  number of digits precision in output table

print  If \texttt{TRUE}, return in print format suitable for \texttt{R Markdown}

newdata  A new data frame of points/observations to apply \texttt{model} to obtain new fitted values and/or predicted values y-hat. Note the format of \texttt{newdata} must match the format of the original data used to fit \texttt{model}.

ID  A string indicating which variable in either the original data used to fit \texttt{model} or \texttt{newdata} should be used as an identification variable to distinguish the observational units in each row. This variable will be the left-most variable in the output data frame. If \texttt{ID} is unspecified, a column \texttt{ID} with values 1 through the number of rows is returned as the identification variable.

Value

A tibble-formatted regression table of outcome/response variable, all explanatory/predictor variables, the fitted/predicted value, and residual.

See Also

\texttt{augment, get_regression_table, get_regression_summaries}

Examples

library(dplyr)
library(tibble)

# Convert rownames to column
mtcars <- mtcars %>%
  rownames_to_column(var = "automobile")

# Fit \texttt{lm()} regression:
mpg_model <- \texttt{lm(mpg \sim cyl, data = mtcars)}

# Get information on all points in regression:
get_regression_points(mpg_model, ID = "automobile")

# Create training and test set based on mtcars:
training_set <- mtcars %>%
  sample_frac(0.5)
test_set <- mtcars %>%
  anti_join(training_set, by = "automobile")

# Fit model to training set:
mpg_model_train <- \texttt{lm(mpg \sim cyl, data = training_set)}

# Make predictions on test set:
get_regression_points(mpg_model_train, newdata = test_set, ID = "automobile")
**get_regression_summaries**

*Get regression summary values*

**Description**

Output scalar summary statistics for an `lm()` regression in “tidy” format. This function is a wrapper function for `broom::glance()`.

**Usage**

```r
get_regression_summaries(model, digits = 3, print = FALSE)
```

**Arguments**

- `model`: an `lm()` model object
- `digits`: number of digits precision in output table
- `print`: If TRUE, return in print format suitable for R Markdown

**Value**

A single-row tibble with regression summaries. Ex: `r_squared` and `mse`.

**See Also**

`glance`, `get_regression_table`, `get_regression_points`

**Examples**

```r
library(moderndive)

# Fit lm() regression:
mpg_model <- lm(mpg ~ cyl, data = mtcars)

# Get regression summaries:
get_regression_summaries(mpg_model)
```
get_regression_table  Get regression table

Description
Output regression table for an \texttt{lm()} regression in "tidy" format. This function is a wrapper function for \texttt{broom::tidy()} and includes confidence intervals in the output table by default.

Usage

\begin{verbatim}
get_regression_table(model, digits = 3, print = FALSE)
\end{verbatim}

Arguments

- \texttt{model} an \texttt{lm()} model object
- \texttt{digits} number of digits precision in output table
- \texttt{print} If \texttt{TRUE}, return in print format suitable for R Markdown

Value
A tibble-formatted regression table along with lower and upper end points of all confidence intervals for all parameters \texttt{lower_ci} and \texttt{upper_ci}.

See Also

\texttt{tidy}, \texttt{get_regression_points}, \texttt{get_regression_summaries}

Examples

\begin{verbatim}
library(moderndive)

# Fit \texttt{lm()} regression:
mpg_model <- \texttt{lm(mpg \sim cyl, data = mtcars)}

# Get regression table:
get_regression_table(mpg_model)
\end{verbatim}
gg_parallel_slopes  

**Plot parallel slopes model**

**Description**

NOTE: THIS FUNCTION WILL SOON BE DEPRECATED; PLEASE USE `{geom_parallel_slopes()}` INSTEAD. Output a visualization of linear regression when you have one numerical and one categorical explanatory/predictor variable: a separate colored regression line for each level of the categorical variable

**Usage**

```r
gg_parallel_slopes(y, num_x, cat_x, data, alpha = 1)
```

**Arguments**

- `y` Character string of outcome variable in `data`
- `num_x` Character string of numerical explanatory/predictor variable in `data`
- `cat_x` Character string of categorical explanatory/predictor variable in `data`
- `data` an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `lm` is called.
- `alpha` Transparency of points

**Value**

A `ggplot` object.

**See Also**

`geom_parallel_slopes`

**Examples**

```r
library(ggplot2)
library(dplyr)
library(moderndive)

# log10() transformations
house_prices <- house_prices %>%
  mutate(
    log10_price = log10(price),
    log10_size = log10(sqft_living)
  )

# Output parallel slopes model plot:
```
gg_parallel_slopes(y = "log10_price", num_x = "log10_size", cat_x = "condition", data = house_prices, alpha = 0.1) +
  labs(x = "log10 square feet living space", y = "log10 price in USD",
       title = "House prices in Seattle: Parallel slopes model")

# Compare with interaction model plot:
ggplot(house_prices, aes(x = log10_size, y = log10_price, col = condition)) +
  geom_point(alpha = 0.1) +
  geom_smooth(method = "lm", se = FALSE, size = 1) +
  labs(x = "log10 square feet living space", y = "log10 price in USD",
       title = "House prices in Seattle: Interaction model")

---

**house_prices**

*House Sales in King County, USA*

**Description**

This dataset contains house sale prices for King County, which includes Seattle. It includes homes sold between May 2014 and May 2015. This dataset was obtained from Kaggle.com [https://www.kaggle.com/harlfoxem/housesalesprediction/data](https://www.kaggle.com/harlfoxem/housesalesprediction/data)

**Usage**

house_prices

**Format**

A data frame with 21613 observations on the following 21 variables.

- **id** a notation for a house
- **date** Date house was sold
- **price** Price is prediction target
- **bedrooms** Number of Bedrooms/House
- **bathrooms** Number of bathrooms/bedrooms
- **sqft_living** square footage of the home
- **sqft_lot** square footage of the lot
- **floors** Total floors (levels) in house
- **waterfront** House which has a view to a waterfront
- **view** Has been viewed
- **condition** How good the condition is (Overall)
- **grade** overall grade given to the housing unit, based on King County grading system
- **sqft_above** square footage of house apart from basement
- **sqft_basement** square footage of the basement
- **yr_built** Built Year
yr_renovated Year when house was renovated
zipcode zip code
lat Latitude coordinate
long Longitude coordinate
sqft_living15 Living room area in 2015 (implies– some renovations) This might or might not have affected the lotsize area
sqft_lot15 lotSize area in 2015 (implies– some renovations)

Source
Kaggle https://www.kaggle.com/harlfoxem/housesalesprediction. Note data is released under a CC0: Public Domain license.

Examples
library(dplyr)
library(ggplot2)

# Create variable log of house price
house_prices <- house_prices %>%
mutate(log_price = log(price))

# Plot histogram of log of house price
ggplot(house_prices, aes(x = log_price)) +
  geom_histogram()

MA_schools Massachusetts Public High Schools Data

Description
Data on Massachusetts public high schools in 2017

Usage
MA_schools

Format
A data frame of 332 rows representing Massachusetts high schools and 4 variables

school_name High school name.
average_sat_math Average SAT math score. Note 58 of the original 390 values of this variable were missing; these rows were dropped from consideration.
perc_disadvan Percent of the student body that are considered economically disadvantaged.
size Size of school enrollment; small 13-341 students, medium 342-541 students, large 542-4264 students.
Source

The original source of the data are Massachusetts Department of Education reports http://profiles.doe.mass.edu/state_report/, however the data was downloaded from Kaggle at https://www.kaggle.com/ndalziel/massachusetts-public-schools-data

Examples

```r
library(ggplot2)
ggplot(MA_schools, aes(x = perc_disadvan, y = average_sat_math, color = size)) + geom_point() + geom_smooth(method = "lm", se = FALSE) + labs(y = "Math SAT score", x = "Percentage economically disadvantaged", color = "School size")
```

moderndive - Tidyverse-Friendly Introductory Linear Regression

Description

Datasets and wrapper functions for tidyverse-friendly introdutory linear regression, used in “Statistical Inference via Data Science: A ModernDive into R and the tidyverse” available at https://moderndive.com/.

Examples

```r
library(moderndive)

# Fit regression model:
mpg_model <- lm(mpg ~ hp, data = mtcars)

# Regression tables:
get_regression_table(mpg_model)

# Information on each point in a regression:
get_regression_points(mpg_model)

# Regression summaries
get_regression_summaries(mpg_model)

# Plotting parallel slopes models
library(ggplot2)
ggplot(evals, aes(x = age, y = score, color = ethnicity)) + geom_point() + geom_parallel_slopes(se = FALSE)
```
movies_sample

Random sample of 68 action and romance movies

Description
A random sample of 32 action movies and 36 romance movies from https://www.imdb.com/ and their ratings.

Usage
movies_sample

Format
A data frame of 68 rows movies.

- title  Movie title
- year  Year released
- rating  IMDb rating out of 10 stars
- genre  Action or Romance

See Also
This data was sampled from the ‘movies’ data frame in movies.

Examples

library(ggplot2)

# Visualize relationship between rating and genre
ggplot(data = movies_sample, aes(x = genre, y = rating)) +
  geom_boxplot() +
  labs(x = "Genre: Action or Romance", y = "IMDb rating")

mythbusters_yawn  Data from Mythbusters’ study on contagiousness of yawning

Description
From a study on whether yawning is contagious https://www.imdb.com/title/tt0768479/. The data here was derived from the final proportions of yawns given in the show.

Usage
mythbusters_yawn
Format

A data frame of 50 rows representing each of the 50 participants in the study.

- **subj**: integer value corresponding to identifier variable of subject ID
- **group**: string of either "seed", participant was shown a yawner, or "control", participant was not shown a yawner
- **yawn**: string of either "yes", the participant yawned, or "no", the participant did not yawn

Examples

```
library(ggplot2)

# Plot both variables as a stacked proportional bar chart
ggplot(mythbusters_yawn, aes(x = group, fill = yawn)) +
  geom_bar(position = "fill") +
  labs(x = "", y = "Proportion",
       title = "Proportion of yawn and not yawn for each group")
```

---

**orig_pennies_sample**  
* A random sample of 40 pennies sampled from the pennies data frame

Description

A dataset of 40 pennies to be treated as a random sample with pennies acting as the population. Data on these pennies were recorded in 2011.

Usage

```
orig_pennies_sample
```

Format

A data frame of 40 rows representing 40 randomly sampled pennies from pennies and 2 variables

- **year**: Year of minting
- **age_in_2011**: Age in 2011

Source


See Also

- pennies
Examples

```r
library(dplyr)
library(ggplot2)

# Take 50 different resamples/bootstraps from the original sample
many_bootstraps <- orig_pennies_sample %>%
  rep_sample_n(size = 40, replace = TRUE, reps = 50)
many_bootstraps

# Compute mean year of minting for each bootstrap sample
bootstrap_means <- many_bootstraps %>%
  group_by(replicate) %>%
  summarize(mean_year = mean(year))

# Plot sampling distribution
ggplot(bootstrap_means, aes(x = mean_year)) +
  geom_histogram(binwidth = 1, color = "white") +
  labs(x = expression(bar(x)), y = "Number of samples",
       title = "Bootstrap distribution of x_bar based 50 resamples of size n = 40")
```

---

**pennies**

* A population of 800 pennies sampled in 2011

---

**Description**

A dataset of 800 pennies to be treated as a sampling population. Data on these pennies were recorded in 2011.

**Usage**

pennies

**Format**

A data frame of 800 rows representing different pennies and 2 variables

- **year**  Year of minting
- **age_in_2011**  Age in 2011

**Source**

Examples

```r
library(dplyr)
library(ggplot2)

# Take 25 different samples of size n = 50 pennies from population
many_samples <- pennies %>%
  rep_sample_n(size = 50, reps = 25)
many_samples

# Compute mean year of minting for each sample
sample_means <- many_samples %>%
  group_by(replicate) %>%
  summarize(mean_year = mean(year))

# Plot sampling distribution
ggplot(sample_means, aes(x = mean_year)) +
  geom_histogram(binwidth = 1, color = "white") +
  labs(x = expression(bar(x)), y = "Number of samples",
       title = "Sampling distribution of x_bar based 25 samples of size n = 50")
```

---

**pennies_resamples**  
*Bootstrap resamples of a sample of 50 pennies*

**Description**

35 bootstrap resamples with replacement of sample of 50 pennies contained in a 50 cent roll from Florence Bank on Friday February 1, 2019 in downtown Northampton, Massachusetts, USA https://goo.gl/maps/AF88fpvVfm12. The original sample of 50 pennies is available in `pennies_sample`.

**Usage**

`pennies_resamples`  

**Format**

A data frame of 1750 rows representing 35 students' bootstrap resamples of size 50 and 3 variables

- **replicate** ID variable of replicate/resample number.
- **name** Name of student
- **year** Year on resampled penny

**See Also**

`pennies_sample`
Examples

```r
library(ggplot2)
library(dplyr)
bootstrap_sample_means <- pennies_resamples %>%
  group_by(name) %>%
  summarize(sample_mean = mean(year))

ggplot(bootstrap_sample_means, aes(x = sample_mean)) +
  geom_histogram(binwidth = 2.5) +
  labs(x = "sample mean year", title = "Bootstrap distribution of sample mean year")
```

---

**pennies_sample**  
A sample of 50 pennies

---

**Description**

A sample of 50 pennies contained in a 50 cent roll from Florence Bank on Friday February 1, 2019 in downtown Northampton, Massachusetts, USA [https://goo.gl/maps/AF88fpvVfm12](https://goo.gl/maps/AF88fpvVfm12).

**Usage**

pennies_sample

**Format**

A data frame of 50 rows representing 50 sampled pennies and 2 variables

- **ID**  Variable used to uniquely identify each penny.
- **year**  Year of minting.

**Note**

The original pennies_sample has been renamed `orig_pennies_sample` as of moderndive v0.3.0.

**Examples**

```r
library(ggplot2)

ggplot(pennies_sample, aes(x = year)) +
  geom_histogram(binwidth = 5, boundary = 2000)
```
**promotions**  
*Bank manager recommendations based on (binary) gender*

**Description**
Data from a 1970’s study on whether gender influences hiring recommendations. Originally used in OpenIntro.org.

**Usage**
```
promotions
```

**Format**
A data frame with 48 observations on the following 3 variables.

- **id**  Identification variable used to distinguish rows.
- **gender** (binary) gender: a factor with two levels 'male' and 'female'
- **decision** a factor with two levels: 'promoted' and 'not'

**Source**

**See Also**
The data in 'gender_promotions' is a tidyverse-friendly transformation of `gender.discrimination`.

**Examples**
```
library(tidyverse)
glimpse(promotions)
```

---

**promotions_shuffled**  
*One permutation/shuffle of promotions*

**Description**
Shuffled/permuted data from a 1970’s study on whether gender influences hiring recommendations.

**Usage**
```
promotions_shuffled
```
**tactile_prop_red**

**Format**

A data frame with 48 observations on the following 3 variables.

- **id** Identification variable used to distinguish rows.
- **gender** shuffled/permutated (binary) gender: a factor with two levels ‘male’ and ‘female’
- **decision** a factor with two levels: ‘promoted’ and ‘not’

**See Also**

promotions.

**Examples**

```r
library(dplyr)
glimpse(promotions)
glimpse(promotions_shuffled)
```

---

**tactile_prop_red**  
* Tactile sampling from a tub of balls

**Description**

Counting the number of red balls in 33 tactile samples of size n = 50 balls from [https://github.com/moderndive/moderndive/blob/master/data-raw/sampling_bowl.jpeg](https://github.com/moderndive/moderndive/blob/master/data-raw/sampling_bowl.jpeg)

**Usage**

```r
tactile_prop_red
```

**Format**

A data frame of 33 rows representing different groups of students’ samples of size n = 50 and 4 variables

- **group** Group members
- **replicate** Replicate number
- **red_balls** Number of red balls sampled out of 50
- **prop_red** Proportion red balls out of 50

**See Also**

bowl
Examples

library(ggplot2)

# Plot sampling distributions
ggplot(tactile_prop_red, aes(x = prop_red)) +
  geom_histogram(binwidth = 0.025) +
  labs(x = expression(hat(p)), y = "Number of samples",
       title = "Sampling distribution of p_hat based 33 samples of size n = 50")
# Index

*Topic **datasets**

- bowl, 2
- bowl_sample_1, 4
- bowl_samples, 3
- DD_vs_SB, 5
- evals, 6
- house_prices, 14
- MA_schools, 15
- movies_sample, 17
- mythbusters_yawn, 17
- orig_pennies_sample, 18
- pennies, 19
- pennies_resamples, 20
- pennies_sample, 21
- promotions, 22
- promotions_shuffled, 22
- tactile_prop_red, 23

- get_regression_table, 10, 11, 12
- gg_parallel_slopes, 13
- ggplot, 13
- ggplot(), 7
- glance, 11
- house_prices, 14
- layer(), 7
- MA_schools, 15
- modernive, 16
- modernive-package (moderndive), 16
- movies, 17
- movies_sample, 17
- mythbusters_yawn, 17
- orig_pennies_sample, 18, 21
- pennies, 18, 19
- pennies_resamples, 20
- pennies_sample, 20, 21
- promotions, 22, 23
- promotions_shuffled, 22
- tactile_prop_red, 23
- tidy, 12

- aes(), 7
- aes_(), 7
- as.data.frame, 13
- augment, 10
- borders(), 8
- bowl, 2, 3, 4, 23
- bowl_sample_1, 4
- bowl_samples, 3
- cor, 9
- DD_vs_SB, 5
- evals, 6
- fortify(), 7
- gender.discrimination, 22
- geom_parallel_slopes, 7, 13
- get_correlation, 8
- get_regression_points, 9, 11, 12
- get_regression_summaries, 10, 11, 12