

Package ‘tidydice’

July 22, 2025

Type Package

Title Simulates Dice Rolls and Coin Flips

Version 1.0.0

Maintainer Roland Krasser <roland.krasser@gmail.com>

Description Utils for basic statistical experiments, that can be used for teaching introductory statistics. Each experiment generates a tibble.
Dice rolls and coin flips are simulated using sample().
The properties of the dice can be changed, like the number of sides.
A coin flip is simulated using a two sided dice.
Experiments can be combined with the pipe-operator.

License GPL-3

Encoding UTF-8

URL <https://github.com/rolkra/tidydice>

Imports assertthat, dplyr, ggplot2, magrittr, purrr, stats, stringr,
tibble, tidyr

RoxygenNote 7.2.1

Suggests explore, knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Author Roland Krasser [aut, cre],
Giovanni Marco Dall'Olio [aut]

Repository CRAN

Date/Publication 2023-02-03 09:40:02 UTC

Contents

binom	2
binom_coin	3
binom_dice	3
circle_points	4

flip_coin	4
force_coin	5
force_dice	6
parse_dice_formula	7
parse_dice_formula_part	7
plot_binom	8
plot_coin	9
plot_dice	10
plot_single_coin	11
plot_single_dice	12
roll_dice	13
roll_dice_formula	14
top_n_dice	15
Index	16

binom	<i>Binomial distribution as table.</i>
-------	--

Description

Generates a tibble containing the binomial distribution using dbinom().

Usage

```
binom(times, prob_success)
```

Arguments

- times number of trials
- prob_success probability of success (number between 0 and 1)

Value

Binomial distribution as a tibble

Examples

```
binom(times = 10, prob_success = 1/10)
```

binom_coin	<i>Binomial distribution of flipping a coin.</i>
------------	--

Description

Generates a tibble containing the binomial distribution of flipping a coin using `dbinom()`.

Usage

```
binom_coin(times, sides = 2, success = 2)
```

Arguments

times	how many times a coin is flipped (or how many coins are flipped at the same time)
sides	number of sides of the coin (default = 2)
success	which result is a success (default = 2)

Value

binomial distribution as a tibble

Examples

```
binom_coin(times = 10)
```

binom_dice	<i>Binomial distribution of rolling a dice.</i>
------------	---

Description

Generates a tibble containing the binomial distribution of rolling the dice using `dbinom()`.

Usage

```
binom_dice(times, sides = 6, success = 6)
```

Arguments

times	How many times a dice is rolled (or how many dice are rolled at the same time)
sides	Number of sides of the dice (default = 6)
success	Which result is a success (default = 6)

Value

Binomial distribution as a tibble

Examples

```
binom_dice(times = 10)
```

circle_points	<i>Helper function to draw a circle</i>
---------------	---

Description

Helper function to draw a circle

Usage

```
circle_points(center = c(0, 0), diameter = 1, npoints = 61)
```

Arguments

center	Vector with x and y coordinate of center
diameter	Diameter of circle
npoints	Number of points used for drawing a circle

Value

Dataframe with x and y coordinates to draw a circle

flip_coin	<i>Simulating flipping a coin.</i>
-----------	------------------------------------

Description

Flipping a coin is simulated using `sample()`. The default coin has 2 sides and is fair. The properties of the coin can be changed. The result is returned as a tibble.

Usage

```
flip_coin(
  data = NULL,
  times = 1,
  rounds = 1,
  success = c(2),
  agg = FALSE,
  sides = 2,
  prob = NULL,
  seed = NULL
)
```

Arguments

data	Data from a previous experiment
times	How many times coin is flipped (or how many coins are flipped at the same time)
rounds	Number of rounds
success	Which result is a success (default = 2)
agg	If TRUE, the result is aggregated (by experiment, rounds)
sides	Number of sides of the coin (default = 2)
prob	Vector of probabilities for each side of the coin
seed	Seed to produce reproducible results

Value

Result of experiment as a tibble

Examples

```
# flipping a coin
flip_coin()

# flipping a coin 10 times
flip_coin(times = 10)

# aggregate result
flip_coin(times = 10, agg = TRUE)

# rounds
flip_coin(times = 10, rounds = 3, agg = TRUE)

# experiments
library(dplyr)
flip_coin(times = 10, rounds = 3, agg = TRUE) %>%
  flip_coin(times = 12, rounds = 3, agg = TRUE)
```

force_coin	<i>Force a coin flipping result.</i>
------------	--------------------------------------

Description

The forced result is returned as a tibble.

Usage

```
force_coin(data = NULL, result = 6, round = 1, experiment = 1, success = 2)
```

Arguments

data	Data from a previous experiment
result	Vector of flipping coin results
round	Round of flipping coin
experiment	Experiment Number
success	Which result is a success (default = 6)

Value

Result of experiment as a tibble

Examples

```
force_coin(6)
force_coin(1:6)
```

force_dice	<i>Force a dice rolling result.</i>
------------	-------------------------------------

Description

The forced result is returned as a tibble.

Usage

```
force_dice(data = NULL, result = 6, round = 1, experiment = 1, success = 6)
```

Arguments

data	Data from a previous experiment
result	Vector of rolling dice results
round	Round of rolling dice
experiment	Experiment Number
success	Which result is a success (default = 6)

Value

Result of experiment as a tibble

Examples

```
force_dice(6)
force_dice(1:6)
```

parse_dice_formula	<i>Given a dice formula string, split it and return a dataframe with the list of functions.</i>
--------------------	---

Description

This is the main function to parse a string containing complex formula specifications for rolling dice.

Usage

```
parse_dice_formula(dice_formula)
```

Arguments

dice_formula A string containing a dice formula, e.g. 1d6e2+1d4

Details

The input can be a string containing specifications for multiple dice, e.g.:

- 1d6e6 -> roll 1 six-sided dice, explode on 6
- 1d6e6+2d4-1d10 -> Roll 1 six-sided dice, explode on 6, plus two 4-sided dice, subtract one 10-sided dice

This is inspired by Avrae's bot syntax for rolling dice. See <https://github.com/avrae/d20>

parse_dice_formula_part	<i>Helper function to parse a dice formula</i>
-------------------------	--

Description

Helper function to parse a dice formula

Usage

```
parse_dice_formula_part(dice_formula_part)
```

Arguments

dice_formula_part A split dice formula, e.g. 1d6e2. For more complex formula, e.g. 1d6e2+3d4, see parse_dice_formula

plot_binom	<i>Plot a binomial distribution.</i>
------------	--------------------------------------

Description

Plot a binomial distribution generated with `binom_dice()` or `binom_coin()`

Usage

```
plot_binom(  
  data,  
  title = "Binomial distribution",  
  color = "darkgrey",  
  color_highlight = "coral",  
  label = NULL,  
  label_size = 3,  
  min_pct = 0.05,  
  highlight = NULL  
)
```

Arguments

<code>data</code>	data containing values for binomial distribution
<code>title</code>	title of the plot
<code>color</code>	color of bars
<code>color_highlight</code>	color of highlighted bars
<code>label</code>	add labels to plot?
<code>label_size</code>	size of label
<code>min_pct</code>	surpress values < min_pct
<code>highlight</code>	vector of values to be highlighted

Value

ggplot object

Examples

```
plot_binom(data = binom_dice(times = 10))
```

plot_coin	<i>Plot result of flip_coin()</i>
-----------	-----------------------------------

Description

Plot result of flip_coin()

Usage

```
plot_coin(  
  data,  
  detailed = FALSE,  
  fill = "white",  
  fill_success = "gold",  
  line_color = "black",  
  line_size = 0.8  
)
```

Arguments

data	result of flip_coin()
detailed	not supported at moment
fill	Fill color
fill_success	Fill color if result is a success
line_color	Color of Lines
line_size	Size of Lines

Value

ggplot-Object

Examples

```
library(magrittr)  
  
# plot one coin  
plot_coin()  
  
# plot multiple coin flips  
flip_coin(times = 3, rounds = 3) %>%  
  plot_coin()  
  
# change coin design  
flip_coin(times = 3, rounds = 3) %>%  
  plot_coin(fill_success = "red")
```

plot_dice	<i>Plot result of roll_dice()</i>
-----------	-----------------------------------

Description

Plot result of roll_dice()

Usage

```
plot_dice(  
  data,  
  detailed = FALSE,  
  fill = "white",  
  fill_success = "gold",  
  point_color = "black",  
  line_color = "black",  
  line_size = 0.8  
)
```

Arguments

data	result of roll_dice()
detailed	If TRUE, the dice is plotted with more details
fill	Fill color
fill_success	Fill color if result is a success
point_color	Color of Points
line_color	Color of Lines
line_size	Size of Lines

Value

ggplot-Object

Examples

```
library(magrittr)  
plot_dice()  
roll_dice(times = 3, rounds = 3) %>% plot_dice()  
roll_dice(times = 3, rounds = 3) %>% plot_dice(fill_success = "red")
```

plot_single_coin	<i>Draw a single coin</i>
------------------	---------------------------

Description

Draw a single coin

Usage

```
plot_single_coin(  
  ggplot = NULL,  
  result = 1,  
  x = 0,  
  y = 0,  
  width = 0.9,  
  fill = "white",  
  detailed = FALSE,  
  line_size = 0.8,  
  line_color = "black"  
)
```

Arguments

ggplot	ggplot-Object. If passed, the dice will be added to plot
result	Result of flip coin (0/1)
x	X-coordinate of dice (center)
y	y-coordinate of dice (center)
width	Width of coin
fill	Fill color
detailed	If TRUE, the dice is plotted with more details
line_size	Size of Lines
line_color	Color of Lines

Value

ggplot-Object

plot_single_dice	<i>Draw a single dice</i>
------------------	---------------------------

Description

Draw a single dice

Usage

```
plot_single_dice(  
  ggplot = NULL,  
  result = 6,  
  x = 0,  
  y = 0,  
  width = 0.9,  
  fill = "white",  
  detailed = FALSE,  
  rounding = width/5,  
  line_size = 0.8,  
  line_color = "black",  
  point_size = width/6,  
  point_color = "black"  
)
```

Arguments

ggplot	ggplot-Object. If passed, the dice will be added to plot
result	Result of dice rolling (0..6)
x	X-coordinate of dice (center)
y	y-coordinate of dice (center)
width	Width of dice
fill	Fill color
detailed	If TRUE, the dice is plotted with more details
rounding	Rounding of dice (only used if detailed == TRUE)
line_size	Size of Lines
line_color	Color of Lines
point_size	Size of Points
point_color	Color of Points

Value

ggplot-Object

roll_dice	<i>Simulating rolling a dice.</i>
-----------	-----------------------------------

Description

Rolling a dice is simulated using `sample()`. The default dice has 6 sides and is fair. The properties of the dice can be changed. The result is returned as a tibble.

Usage

```
roll_dice(  
  data = NULL,  
  times = 1,  
  rounds = 1,  
  success = c(6),  
  agg = FALSE,  
  sides = 6,  
  prob = NULL,  
  seed = NULL  
)
```

Arguments

<code>data</code>	Data from a previous experiment
<code>times</code>	How many times a dice is rolled (or how many dice are rolled at the same time)
<code>rounds</code>	Number of rounds
<code>success</code>	Which result is a success (default = 6)
<code>agg</code>	If TRUE, the result is aggregated (by experiment, rounds)
<code>sides</code>	Number of sides of the dice (default = 6)
<code>prob</code>	Vector of probabilities for each side of the dice
<code>seed</code>	Seed to produce reproducible results

Value

Result of experiment as a tibble

Examples

```
# rolling a dice once  
roll_dice()  
  
# rolling a dice 10 times  
roll_dice(times = 10)  
  
# aggregate result  
roll_dice(times = 10, agg = TRUE)
```

```
# rounds
roll_dice(times = 10, rounds = 3, agg = TRUE)

# experiments
library(dplyr)
roll_dice(times = 10, rounds = 3, agg = TRUE) %>%
  roll_dice(times = 12, rounds = 3, agg = TRUE)
```

roll_dice_formula	<i>Simulating rolling a dice, using a formula</i>
-------------------	---

Description

Simulating rolling a dice, using a formula

Usage

```
roll_dice_formula(
  data = NULL,
  dice_formula = "1d6",
  times = 1,
  rounds = 1,
  seed = NULL,
  prob = NULL,
  success = c(6),
  agg = FALSE,
  label = NULL
)
```

Arguments

data	Data from a previous experiment
dice_formula	Dice formula (e.g. "1d6" = 1 dice with 6 sides)
times	How many times a dice is rolled (or how many dice are rolled at the same time)
rounds	Number of rounds
seed	Seed to produce reproducible results
prob	Vector of probabilities for each side of the dice
success	Which result is a success (default = 6)
agg	If TRUE, the result is aggregated (by experiment, rounds) (not implemented)
label	Custom text to distinguish an experiment, can be used for plotting etc.

Value

Result of experiment as a tibble

Examples

```
# roll one 6-sided dice
roll_dice_formula(dice_formula = "1d6")

# roll one 8-sided dice
roll_dice_formula(dice_formula = "1d8")

# roll two 6-sided dice
roll_dice_formula(dice_formula = "2d6")

# roll two 6-sided dice, explode dice on a 6
roll_dice_formula(dice_formula = "2d6e6")

# roll three 6-sided dice, keep highest 2 rolls
roll_dice_formula(dice_formula = "3d6kh2")

# roll three 6-sided dice, keep lowest 2 rolls
roll_dice_formula(dice_formula = "3d6kl2")

# roll four 6-sided dice, keep highest 3 rolls, but explode on a 6
roll_dice_formula(dice_formula = "4d6kh3e6")

# roll one 20-sided dice, and add 4
roll_dice_formula(dice_formula = "1d20+4")

# roll one 4-sided dice and one 6-sided dice, and sum the results
roll_dice_formula(dice_formula = "1d4+1d6")
```

top_n_dice*Helper function to get sum of top n dice*

Description

Helper function to get sum of top n dice

Usage

```
top_n_dice(x, n, dec = F)
```

Arguments

x	Vector of dice-values
n	Number of dice
dec	Decreasing

Index

`binom`, [2](#)
`binom_coin`, [3](#)
`binom_dice`, [3](#)

`circle_points`, [4](#)

`flip_coin`, [4](#)
`force_coin`, [5](#)
`force_dice`, [6](#)

`parse_dice_formula`, [7](#)
`parse_dice_formula_part`, [7](#)
`plot_binom`, [8](#)
`plot_coin`, [9](#)
`plot_dice`, [10](#)
`plot_single_coin`, [11](#)
`plot_single_dice`, [12](#)

`roll_dice`, [13](#)
`roll_dice_formula`, [14](#)

`top_n_dice`, [15](#)