

# Package ‘descriptr’

February 1, 2020

**Type** Package

**Title** Generate Descriptive Statistics

**Version** 0.5.1

**Description** Generate descriptive statistics such as measures of location, dispersion, frequency tables, cross tables, group summaries and multiple one/two way tables.

**Depends** R(>= 3.2)

**Imports** cli, dplyr, ggplot2, graphics, gridExtra, magrittr, purrr, rlang, scales, stats, tibble, tidyr, utils, vistributions, xplorerr

**Suggests** covr, knitr, rmarkdown, testthat, vdiff

**License** MIT + file LICENSE

**URL** <https://descriptr.rsquaredacademy.com/>,  
<https://github.com/rsquaredacademy/descriptr>

**BugReports** <https://github.com/rsquaredacademy/descriptr/issues>

**Encoding** UTF-8

**LazyData** true

**VignetteBuilder** knitr

**RoxygenNote** 6.1.1

**NeedsCompilation** no

**Author** Aravind Hebbali [aut, cre] (<<https://orcid.org/0000-0001-9220-9669>>)

**Maintainer** Aravind Hebbali <[hebbali.aravind@gmail.com](mailto:hebbali.aravind@gmail.com)>

**Repository** CRAN

**Date/Publication** 2020-02-01 08:20:02 UTC

**R topics documented:**

descriptr . . . . .	3
dist_binom_plot . . . . .	3
dist_chi_plot . . . . .	4
dist_f_plot . . . . .	5
dist_norm_plot . . . . .	6
dist_t . . . . .	7
ds_auto_freq_table . . . . .	8
ds_auto_group_summary . . . . .	9
ds_auto_summary_stats . . . . .	10
ds_cross_table . . . . .	10
ds_css . . . . .	11
ds_cvar . . . . .	12
ds_extreme_obs . . . . .	12
ds_freq_table . . . . .	13
ds_gmean . . . . .	14
ds_group_summary . . . . .	14
ds_group_summary_interact . . . . .	15
ds_hmean . . . . .	16
ds_kurtosis . . . . .	17
ds_launch_shiny_app . . . . .	18
ds_mdev . . . . .	18
ds_measures_location . . . . .	19
ds_measures_symmetry . . . . .	20
ds_measures_variation . . . . .	20
ds_mode . . . . .	21
ds_percentiles . . . . .	21
ds_plot_bar . . . . .	22
ds_plot_bar_grouped . . . . .	23
ds_plot_bar_stacked . . . . .	23
ds_plot_box_group . . . . .	24
ds_plot_box_single . . . . .	24
ds_plot_density . . . . .	25
ds_plot_histogram . . . . .	25
ds_plot_scatter . . . . .	26
ds_range . . . . .	26
ds_rindex . . . . .	27
ds_screener . . . . .	28
ds_skewness . . . . .	29
ds_std_error . . . . .	30
ds_summary_stats . . . . .	30
ds_tailobs . . . . .	31
ds_tidy_stats . . . . .	31
hsb . . . . .	32
mtcarz . . . . .	33

---

descriptr	descriptr <i>package</i>
-----------	--------------------------

---

**Description**

Generate descriptive statistics and explore statistical distributions

---

dist_binom_plot	<i>Visualize binomial distribution</i>
-----------------	--

---

**Description**

Visualize how changes in number of trials and the probability of success affect the shape of the binomial distribution. Compute & visualize probability from a given quantile and quantiles out of given probability.

**Usage**

```
dist_binom_plot(n, p)
```

```
dist_binom_prob(n, p, s, type = c("lower", "upper", "exact", "interval"))
```

```
dist_binom_perc(n, p, tp, type = c("lower", "upper"))
```

**Arguments**

n	Number of trials.
p	Aggregate probability.
s	Number of success.
type	Lower/upper/exact/interval.
tp	Probability of success in a trial.

**Value**

A list containing the following components:

avg	Mean of the binomial distribution,
stdev	Standard deviation of the binomial distribution.
prob	Probability of success.

**See Also**

[Binomial](#)

**Examples**

```
# visualize binomial distribution
dist_binom_plot(10, 0.3)

# visualize probability from a given quantile
dist_binom_prob(10, 0.3, 4, type = 'exact')
dist_binom_prob(10, 0.3, 4, type = 'lower')
dist_binom_prob(10, 0.3, 4, type = 'upper')
dist_binom_prob(10, 0.3, c(4, 6), type = 'interval')

# visualize quantiles out of given probability
dist_binom_perc(10, 0.5, 0.05)
dist_binom_perc(10, 0.5, 0.05, "upper")
```

---

dist\_chi\_plot

*Visualize chi square distribution*


---

**Description**

Visualize how changes in degrees of freedom affect the shape of the chi square distribution. Compute & visualize quantiles out of given probability and probability from a given quantile.

**Usage**

```
dist_chi_plot(df = 3, normal = FALSE)

dist_chi_perc(probs = 0.95, df = 3, type = c("lower", "upper"))

dist_chi_prob(perc, df, type = c("lower", "upper"))
```

**Arguments**

df	Degrees of freedom.
normal	If TRUE, normal curve with same mean and sd as the chi square distribution is drawn.
probs	Probability value.
type	Lower tail or upper tail.
perc	Quantile value.

**Value**

Percentile for the probs based on df and type or probability value for perc based on df and type.

**See Also**

[Chisquare](#)

**Examples**

```
# visualize chi square distribution
dist_chi_plot()
dist_chi_plot(df = 5)
dist_chi_plot(df = 5, normal = TRUE)

# visualize quantiles out of given probability
dist_chi_perc(0.165, 8, 'upper')
dist_chi_perc(0.22, 13, 'upper')

# visualize probability from a given quantile.
dist_chi_prob(13.58, 11, 'lower')
dist_chi_prob(15.72, 13, 'upper')
```

---

`dist_f_plot`*Visualize f distribution*

---

**Description**

Visualize how changes in degrees of freedom affect the shape of the F distribution. Compute & visualize quantiles out of given probability and probability from a given quantile.

**Usage**

```
dist_f_plot(num_df = 4, den_df = 30, normal = FALSE)

dist_f_perc(probs = 0.95, num_df = 3, den_df = 30,
            type = c("lower", "upper"))

dist_f_prob(perc, num_df, den_df, type = c("lower", "upper"))
```

**Arguments**

<code>num_df</code>	Degrees of freedom associated with the numerator of f statistic.
<code>den_df</code>	Degrees of freedom associated with the denominator of f statistic.
<code>normal</code>	If TRUE, normal curve with same mean and sd as the F distribution is drawn.
<code>probs</code>	Probability value.
<code>type</code>	Lower tail or upper tail.
<code>perc</code>	Quantile value.

**Value**

Percentile for the `probs` based on `num_df`, `den_df` and `type` or probability value for `perc` based on `num_df`, `den_df` and `type`.

**See Also**[FDist](#)**Examples**

```
# visualize F distribution
dist_f_plot()
dist_f_plot(6, 10, normal = TRUE)

# visualize probability from a given quantile
dist_f_perc(0.95, 3, 30, 'lower')
dist_f_perc(0.125, 9, 35, 'upper')

# visualize quantiles out of given probability
dist_f_prob(2.35, 5, 32)
dist_f_prob(1.5222, 9, 35, type = "upper")
```

---

dist_norm_plot	<i>Visualize normal distribution</i>
----------------	--------------------------------------

---

**Description**

Visualize how changes in mean and standard deviation affect the shape of the normal distribution. Compute & visualize quantiles out of given probability and probability from a given quantile.

**Usage**

```
dist_norm_plot(mean = 0, sd = 1)

dist_norm_perc(probs = 0.95, mean = 0, sd = 1, type = c("lower",
  "upper", "both"))

dist_norm_prob(perc, mean = 0, sd = 1, type = c("lower", "upper",
  "both"))
```

**Arguments**

mean	Mean of the normal distribution.
sd	Standard deviation of the normal distribution.
probs	Probability value.
type	Lower tail, upper tail or both.
perc	Quantile value.

**Value**

Percentile for the probs based on mean, sd and type or probability value for perc based on mean, sd and type.

**See Also**[Normal](#)**Examples**

```
# visualize normal distribution
dist_norm_plot()
dist_norm_plot(mean = 2, sd = 0.6)

# visualize probability from a given quantile
dist_norm_prob(3.78, mean = 2, sd = 1.36)
dist_norm_prob(3.43, mean = 2, sd = 1.36, type = 'upper')
dist_norm_prob(c(-1.74, 1.83), type = 'both')

# visualize quantiles out of given probability
dist_norm_perc(0.95, mean = 2, sd = 1.36)
dist_norm_perc(0.3, mean = 2, sd = 1.36, type = 'upper')
dist_norm_perc(0.95, mean = 2, sd = 1.36, type = 'both')
```

---

`dist_t`*Visualize t distribution*

---

**Description**

Visualize how degrees of freedom affect the shape of t distribution, visualize quantiles out of given probability and probability from a given quantile.

**Usage**

```
dist_t_plot(df = 3)

dist_t_perc(probs = 0.95, df = 4, type = c("lower", "upper", "both"))

dist_t_prob(perc, df, type = c("lower", "upper", "interval", "both"))
```

**Arguments**

<code>df</code>	Degrees of freedom.
<code>probs</code>	Probability value.
<code>type</code>	Lower tail, upper tail, interval or both.
<code>perc</code>	Quantile value.

**Value**

Percentile for the `probs` based on `df` and `type` or probability value for the `perc` based on `df` and `type`.

**See Also**[TDist](#)**Examples**

```
# visualize t distribution
dist_t_plot()
dist_t_plot(6)
dist_t_plot(df = 8)

# visualize quantiles out of given probability
dist_t_perc(probs = 0.95, df = 4, type = 'lower')
dist_t_perc(probs = 0.35, df = 4, type = 'upper')
dist_t_perc(probs = 0.69, df = 7, type = 'both')

# visualize probability from a given quantile
dist_t_prob(2.045, 7, 'lower')
dist_t_prob(0.945, 7, 'upper')
dist_t_prob(1.445, 7, 'interval')
dist_t_prob(1.6, 7, 'both')
```

---

ds\_auto\_freq\_table      *Multiple One & Two Way Tables*

---

**Description**

ds\_auto\_freq\_table creates multiple one way tables by creating a frequency table for each categorical variable in a data frame. ds\_auto\_cross\_table creates multiple two way tables by creating a cross table for each unique pair of categorical variables in a data frame.

**Usage**

```
ds_auto_freq_table(data, ...)
```

```
ds_auto_cross_table(data, ...)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.

**Details**

ds\_auto\_freq\_table is an extension of the ds\_freq\_table function. It creates a frequency table for each categorical variable in the dataframe. ds\_auto\_cross\_table is an extension of the ds\_cross\_table function. It creates a two way table for each unique pair of categorical variables in the dataframe.



## Deprecated Functions

ds\_oway\_tables() and ds\_tway\_tables() have been deprecated. Instead use ds\_auto\_freq\_table() and ds\_auto\_cross\_table().

## See Also

[link{ds\\_freq\\_table}](#) [link{ds\\_cross\\_table}](#)

## Examples

```
# multiple one way tables
ds_auto_freq_table(mtcars)
ds_auto_freq_table(mtcars, cyl, gear)

# multiple two way tables
ds_auto_cross_table(mtcars)
ds_auto_cross_table(mtcars, cyl, gear, am)
```

---

ds\_auto\_group\_summary *Tabulation*

---

## Description

Generate summary statistics for all continuous variables in data.

## Usage

```
ds_auto_group_summary(data, ...)
```

## Arguments

data	A data.frame or tibble.
...	Column(s) in data.

## Examples

```
ds_auto_group_summary(mtcars, cyl, gear, mpg, disp)
```

---

ds\_auto\_summary\_stats *Descriptive statistics and frequency tables*

---

### Description

Generate summary statistics & frequency table for all continuous variables in data.

### Usage

```
ds_auto_summary_stats(data, ...)
```

### Arguments

data	A data.frame or tibble.
...	Column(s) in data.

### Examples

```
ds_auto_summary_stats(mtcars)
ds_auto_summary_stats(mtcars, disp, hp)
```

---

ds\_cross\_table *Two way table*

---

### Description

Creates two way tables of categorical variables. The tables created can be visualized as barplots and mosaicplots.

### Usage

```
ds_cross_table(data, var1, var2)

## S3 method for class 'ds_cross_table'
plot(x, stacked = FALSE, proportional = FALSE,
     print_plot = TRUE, ...)

ds_twoway_table(data, var1, var2)
```

**Arguments**

data	A data.frame or a tibble.
var1	First categorical variable.
var2	Second categorical variable.
x	An object of class cross_table.
stacked	If FALSE, the columns of height are portrayed as stacked bars, and if TRUE the columns are portrayed as juxtaposed bars.
proportional	If TRUE, the height of the bars is proportional.
print_plot	logical; if TRUE, prints the plot else returns a plot object.
...	Further arguments to be passed to or from methods.

**Examples**

```
k <- ds_cross_table(mtcars, cyl, gear)
k

# bar plots
plot(k)
plot(k, stacked = TRUE)
plot(k, proportional = TRUE)

# alternate
ds_twoway_table(mtcars, cyl, gear)
```

---

ds_css	<i>Corrected Sum of Squares</i>
--------	---------------------------------

---

**Description**

Compute the corrected sum of squares

**Usage**

```
ds_css(x, na.rm = FALSE)
```

**Arguments**

x	a numeric vector containing the values whose mode is to be computed
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

**Details**

Any NA values are stripped from x before computation takes place.

**Value**

Corrected sum of squares of x

**Examples**

```
ds_css(mtcars$mpg)
```

---

ds_cvar	<i>Coefficient of Variation</i>
---------	---------------------------------

---

**Description**

Compute the coefficient of variation

**Usage**

```
ds_cvar(x, na.rm = FALSE)
```

**Arguments**

x	a numeric vector containing the values whose mode is to be computed
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

**Details**

Any NA values are stripped from x before computation takes place.

**Examples**

```
ds_cvar(mtcars$mpg)
```

---

ds_extreme_obs	<i>Extreme observations</i>
----------------	-----------------------------

---

**Description**

Returns the most extreme observations.

**Usage**

```
ds_extreme_obs(data, column)
```

**Arguments**

data	A data.frame or tibble.
column	Column in data.

**Examples**

```
ds_extreme_obs(mtcars, mpg)
```

---

ds_freq_table	<i>Frequency table</i>
---------------	------------------------

---

**Description**

Frequency table for categorical and continuous data and returns the frequency, cumulative frequency, frequency percent and cumulative frequency percent. `plot.ds_freq_table()` creates bar plot for the categorical data and histogram for continuous data.

**Usage**

```
ds_freq_table(data, variable, bins = 5)
```

```
## S3 method for class 'ds_freq_table'  
plot(x, print_plot = TRUE, ...)
```

**Arguments**

data	A data.frame or a tibble.
variable	Column in data.
bins	Number of intervals into which the data must be split.
x	An object of class ds_freq_table.
print_plot	logical; if TRUE, prints the plot else returns a plot object.
...	Further arguments to be passed to or from methods.

**See Also**

[ds\\_cross\\_table](#)

**Examples**

```
# categorical data  
ds_freq_table(mtcars, cyl)  
  
# barplot  
k <- ds_freq_table(mtcars, cyl)  
plot(k)  
  
# continuous data  
ds_freq_table(mtcars, mpg)  
  
# barplot  
k <- ds_freq_table(mtcars, mpg)  
plot(k)
```

---

ds_gmean	<i>Geometric Mean</i>
----------	-----------------------

---

**Description**

Compute the geometric mean

**Usage**

```
ds_gmean(x, na.rm = FALSE, ...)
```

**Arguments**

x	a numeric vector containing the values whose geometric mean is to be computed
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
...	further arguments passed to or from other methods #' @details Any NA values are stripped from x before computation takes place.

**Value**

Returns the geometric mean of x

**See Also**

[ds\\_hmean mean](#)

**Examples**

```
ds_gmean(mtcars$mpg)
```

---

ds_group_summary	<i>Groupwise descriptive statistics</i>
------------------	---

---

**Description**

Descriptive statistics of a continuous variable for the different levels of a categorical variable. `boxplot.group_summary()` creates boxplots of the continuous variable for the different levels of the categorical variable.

**Usage**

```
ds_group_summary(data, gvar, cvar)
```

```
## S3 method for class 'ds_group_summary'
plot(x, print_plot = TRUE, ...)
```

**Arguments**

data	A data.frame or a tibble.
gvar	Column in data.
cvar	Column in data.
x	An object of the class ds_group_summary.
print_plot	logical; if TRUE, prints the plot else returns a plot object.
...	Further arguments to be passed to or from methods.

**Value**

ds\_group\_summary() returns an object of class "ds\_group\_summary". An object of class "ds\_group\_summary" is a list containing the following components:

stats	A data frame containing descriptive statistics for the different levels of the factor variable.
tidy_stats	A tibble containing descriptive statistics for the different levels of the factor variable.
plotdata	Data for boxplot method.

**See Also**

[ds\\_summary\\_stats](#)

**Examples**

```
# ds_group summary
ds_group_summary(mtcars, cyl, mpg)

# boxplot
k <- ds_group_summary(mtcars, cyl, mpg)
plot(k)

# tibble
k$tidy_stats
```

---

ds\_group\_summary\_interact

*Category wise descriptive statistics*

---

**Description**

Descriptive statistics of a continuous variable for the combination of levels of two or more categorical variables.

**Usage**

```
ds_group_summary_interact(data, cvar, ...)
```

**Arguments**

data	A data.frame or a tibble.
cvar	Column in data; continuous variable.
...	Columns in data; categorical variables.

**See Also**

[ds\\_group\\_summary](#)

**Examples**

```
ds_group_summary_interact(mtcars, mpg, cyl, gear)
```

---

ds_hmean	<i>Harmonic Mean</i>
----------	----------------------

---

**Description**

Compute the harmonic mean

**Usage**

```
ds_hmean(x, na.rm = FALSE, ...)
```

**Arguments**

x	a numeric vector containing the values whose harmonic mean is to be computed
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
...	further arguments passed to or from other methods #' @details Any NA values are stripped from x before computation takes place.

**Value**

Returns the harmonic mean of x

**See Also**

[ds\\_gmean mean](#)

**Examples**

```
ds_hmean(mtcars$mpg)
```



---

ds_kurtosis	<i>Kurtosis</i>
-------------	-----------------

---

**Description**

Compute the kurtosis of a probability distribution.

**Usage**

```
ds_kurtosis(x, na.rm = FALSE)
```

**Arguments**

x	a numeric vector containing the values whose kurtosis is to be computed
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

**Details**

Any NA values are stripped from x before computation takes place.

**Value**

Kurtosis of x

**References**

Sheskin, D.J. (2000) Handbook of Parametric and Nonparametric Statistical Procedures, Second Edition. Boca Raton, Florida: Chapman & Hall/CRC.

**See Also**

ds\_skewness

**Examples**

```
ds_kurtosis(mtcars$mpg)
```

ds\_launch\_shiny\_app     *Launch Shiny App*

---

**Description**

Launches shiny app

**Usage**

```
ds_launch_shiny_app()
```

**Deprecated Function**

launch\_descriptr() has been deprecated. Instead use ds\_launch\_shiny\_app().

**Examples**

```
## Not run:  
ds_launch_shiny_app()  
  
## End(Not run)
```

---

ds\_mdev     *Mean Absolute Deviation*

---

**Description**

Compute the mean absolute deviation about the mean

**Usage**

```
ds_mdev(x, na.rm = FALSE)
```

**Arguments**

x	a numeric vector
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

**Details**

The stat\_mdev function computes the mean absolute deviation about the mean. It is different from mad in stats package as the statistic used to compute the deviations is not median but mean. Any NA values are stripped from x before computation takes place

**Value**

Mean absolute deviation of x

**See Also**

[mad](#)

**Examples**

```
ds_mdev(mtcars$mpg)
```

---

ds\_measures\_location *Measures of location*

---

**Description**

Returns the measures of location such as mean, median & mode.

**Usage**

```
ds_measures_location(data, ..., trim = 0.05)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
trim	The fraction of values to be trimmed before computing the mean.

**Examples**

```
ds_measures_location(mtcars)
ds_measures_location(mtcars, mpg)
ds_measures_location(mtcars, mpg, disp)
```

ds\_measures\_symmetry *Measures of symmetry*

---

**Description**

Returns the measures of symmetry such as skewness and kurtosis.

**Usage**

```
ds_measures_symmetry(data, ...)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.

**Examples**

```
ds_measures_symmetry(mtcars)
ds_measures_symmetry(mtcars, mpg)
ds_measures_symmetry(mtcars, mpg, disp)
```

---

ds\_measures\_variation *Measures of variation*

---

**Description**

Returns the measures of location such as range, variance and standard deviation.

**Usage**

```
ds_measures_variation(data, ...)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.

**Examples**

```
ds_measures_variation(mtcars)
ds_measures_variation(mtcars, mpg)
ds_measures_variation(mtcars, mpg, disp)
```

---

ds_mode	<i>Mode</i>
---------	-------------

---

**Description**

Compute the sample mode

**Usage**

```
ds_mode(x, na.rm = FALSE)
```

**Arguments**

x	a numeric vector containing the values whose mode is to be computed
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

**Details**

Any NA values are stripped from x before computation takes place.

**Value**

Mode of x

**See Also**

[mean](#) [median](#)

**Examples**

```
ds_mode(mtcars$mpg)
ds_mode(mtcars$cyl)
```

---

ds_percentiles	<i>Percentiles</i>
----------------	--------------------

---

**Description**

Returns the percentiles

**Usage**

```
ds_percentiles(data, ...)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.

**Examples**

```
ds_percentiles(mtcars)
ds_percentiles(mtcars, mpg)
ds_percentiles(mtcars, mpg, disp)
```

---

ds_plot_bar	<i>Generate bar plots</i>
-------------	---------------------------

---

**Description**

Creates bar plots if the data has categorical variables.

**Usage**

```
ds_plot_bar(data, ..., fill = "blue", print_plot = TRUE)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
fill	Color of the bars.
print_plot	logical; if TRUE, prints the plot else returns a plot object.

**Examples**

```
ds_plot_bar(mtcars)
ds_plot_bar(mtcars, cyl)
ds_plot_bar(mtcars, cyl, gear)
```

---

ds\_plot\_bar\_grouped     *Generate grouped bar plots*

---

**Description**

Creates grouped bar plots if the data has categorical variables.

**Usage**

```
ds_plot_bar_grouped(data, ..., print_plot = TRUE)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
print_plot	logical; if TRUE, prints the plot else returns a plot object.

**Examples**

```
mt <- dplyr::select(mtcars, cyl, gear, am)
ds_plot_bar_grouped(mt)
ds_plot_bar_grouped(mtcars, cyl, gear)
```

---

ds\_plot\_bar\_stacked     *Generate stacked bar plots*

---

**Description**

Creates stacked bar plots if the data has categorical variables.

**Usage**

```
ds_plot_bar_stacked(data, ..., print_plot = TRUE)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
print_plot	logical; if TRUE, prints the plot else returns a plot object.

**Examples**

```
mt <- dplyr::select(mtcars, cyl, gear, am)
ds_plot_bar_stacked(mt)
ds_plot_bar_stacked(mtcars, cyl, gear)
```

---

ds\_plot\_box\_group      *Compare distributions*

---

**Description**

Creates box plots if the data has both categorical & continuous variables.

**Usage**

```
ds_plot_box_group(data, ..., print_plot = TRUE)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
print_plot	logical; if TRUE, prints the plot else returns a plot object.

**Examples**

```
mt <- dplyr::select(mtcars, cyl, disp, mpg)
ds_plot_box_group(mt)
ds_plot_box_group(mtcars, cyl, gear, mpg)
```

---

ds\_plot\_box\_single      *Generate box plots*

---

**Description**

Creates box plots if the data has continuous variables.

**Usage**

```
ds_plot_box_single(data, ..., print_plot = TRUE)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
print_plot	logical; if TRUE, prints the plot else returns a plot object.

**Examples**

```
ds_plot_box_single(mtcars)
ds_plot_box_single(mtcars, mpg)
ds_plot_box_single(mtcars, mpg, disp, hp)
```



---

ds_plot_density	<i>Generate density plots</i>
-----------------	-------------------------------

---

**Description**

Creates density plots if the data has continuous variables.

**Usage**

```
ds_plot_density(data, ..., color = "blue", print_plot = TRUE)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
color	Color of the plot.
print_plot	logical; if TRUE, prints the plot else returns a plot object.

**Examples**

```
ds_plot_density(mtcars)
ds_plot_density(mtcars, mpg)
ds_plot_density(mtcars, mpg, disp, hp)
```

---

ds_plot_histogram	<i>Generate histograms</i>
-------------------	----------------------------

---

**Description**

Creates histograms if the data has continuous variables.

**Usage**

```
ds_plot_histogram(data, ..., bins = 5, fill = "blue",
  print_plot = TRUE)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
bins	Number of bins in the histogram.
fill	Color of the histogram.
print_plot	logical; if TRUE, prints the plot else returns a plot object.

**Examples**

```
ds_plot_histogram(mtcars)
ds_plot_histogram(mtcars, mpg)
ds_plot_histogram(mtcars, mpg, disp, hp)
```

---

ds_plot_scatter	<i>Generate scatter plots</i>
-----------------	-------------------------------

---

**Description**

Creates scatter plots if the data has continuous variables.

**Usage**

```
ds_plot_scatter(data, ..., print_plot = TRUE)
```

**Arguments**

data	A data.frame or tibble.
...	Column(s) in data.
print_plot	logical; if TRUE, prints the plot else returns a plot object.

**Examples**

```
ds_plot_scatter(mtcars)
ds_plot_scatter(mtcars, mpg, disp)
```

---

ds_range	<i>Range</i>
----------	--------------

---

**Description**

Compute the range of a numeric vector

**Usage**

```
ds_range(x, na.rm = FALSE)
```

**Arguments**

x	a numeric vector
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

**Value**

Range of x

**See Also**

[range](#)

**Examples**

```
ds_range(mtcars$mpg)
```

---

ds_rindex	<i>Index Values</i>
-----------	---------------------

---

**Description**

Returns index of values.

**Usage**

```
ds_rindex(data, values)
```

**Arguments**

data	a numeric vector
values	a numeric vector containing the values whose index is returned

**Details**

Any NA values are stripped from data and values before computation takes place.

**Value**

Index of the values in data. In case, data does not contain index, NULL is returned.

**Examples**

```
ds_rindex(mtcars$mpg, 21)  
ds_rindex(mtcars$mpg, 22)
```

---

ds_screener	<i>Screen data</i>
-------------	--------------------

---

### Description

Screen data and return details such as variable names, class, levels and missing values. `plot.ds_screener()` creates bar plots to visualize of missing observations for each variable in a data set.

### Usage

```
ds_screener(data)

## S3 method for class 'ds_screener'
plot(x, ...)
```

### Arguments

<code>data</code>	A tibble or a data.frame.
<code>x</code>	An object of class <code>ds_screener</code> .
<code>...</code>	Further arguments to be passed to or from methods.

### Value

`ds_screener()` returns an object of class "ds\_screener". An object of class "ds\_screener" is a list containing the following components:

<code>Rows</code>	Number of rows in the data frame.
<code>Columns</code>	Number of columns in the data frame.
<code>Variables</code>	Names of the variables in the data frame.
<code>Types</code>	Class of the variables in the data frame.
<code>Count</code>	Length of the variables in the data frame.
<code>nlevels</code>	Number of levels of a factor variable.
<code>levels</code>	Levels of factor variables in the data frame.
<code>Missing</code>	Number of missing observations in each variable.
<code>MissingPer</code>	Percent of missing observations in each variable.
<code>MissingTotal</code>	Total number of missing observations in the data frame.
<code>MissingTotPer</code>	Total percent of missing observations in the data frame.
<code>MissingRows</code>	Total number of rows with missing observations in the data frame.
<code>MissingCols</code>	Total number of columns with missing observations in the data frame.

### Examples

```
# screen data
ds_screener(mtcars)
```

---

ds_skewness	<i>Skewness</i>
-------------	-----------------

---

**Description**

Compute the skewness of a probability distribution.

**Usage**

```
ds_skewness(x, na.rm = FALSE)
```

**Arguments**

x	a numeric vector containing the values whose skewness is to be computed
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

**Details**

Any NA values are stripped from x before computation takes place.

**Value**

Skewness of x

**References**

Sheskin, D.J. (2000) Handbook of Parametric and Nonparametric Statistical Procedures, Second Edition. Boca Raton, Florida: Chapman & Hall/CRC.

**See Also**

kurtosis

**Examples**

```
ds_skewness(mtcars$mpg)
```

---

ds_std_error	<i>Standard error of mean</i>
--------------	-------------------------------

---

**Description**

Returns the standard error of mean.

**Usage**

```
ds_std_error(x)
```

**Arguments**

x                    A numeric vector.

**Examples**

```
ds_std_error(mtcars$mpg)
```

---

ds_summary_stats	<i>Descriptive statistics</i>
------------------	-------------------------------

---

**Description**

Range of descriptive statistics for continuous data.

**Usage**

```
ds_summary_stats(data, ...)
```

**Arguments**

data                A data.frame or tibble.  
...                 Column(s) in data.

**See Also**

[summary](#) [ds\\_freq\\_table](#) [ds\\_cross\\_table](#)

**Examples**

```
ds_summary_stats(mtcars, mpg)
```

---

ds_tailobs	<i>Tail Observations</i>
------------	--------------------------

---

**Description**

Returns the n highest/lowest observations from a numeric vector.

**Usage**

```
ds_tailobs(data, n, type = c("low", "high"))
```

**Arguments**

data	a numeric vector
n	number of observations to be returned
type	if low, the n lowest observations are returned, else the highest n observations are returned

**Details**

Any NA values are stripped from data before computation takes place.

**Value**

n highest/lowest observations from data

**See Also**

[top\\_n](#)

**Examples**

```
ds_tailobs(mtcars$mpg, 5)  
ds_tailobs(mtcars$mpg, 5, type = "high")
```

---

ds_tidy_stats	<i>Tidy descriptive statistics</i>
---------------	------------------------------------

---

**Description**

Descriptive statistics for multiple variables.

**Usage**

```
ds_tidy_stats(data, ...)
```

**Arguments**

`data` A tibble or a `data.frame`.  
`...` Columns in `x`.

**Value**

A tibble.

**Deprecated Functions**

`ds_multi_stats()` have been deprecated. Instead use `ds_tidy_stats()`.

**Examples**

```
ds_tidy_stats(mtcars)
ds_tidy_stats(mtcars, mpg, disp, hp)
```

---

hsb

*High School and Beyond Data Set*

---

**Description**

A dataset containing demographic information and standardized test scores of high school students.

**Usage**

hsb

**Format**

A data frame with 200 rows and 10 variables:

**id** id of the student  
**female** gender of the student  
**race** ethnic background of the student  
**ses** socio-economic status of the student  
**schtyp** school type  
**prog** program type  
**read** scores from test of reading  
**write** scores from test of writing  
**math** scores from test of math  
**science** scores from test of science  
**socst** scores from test of social studies



**Source**

<http://www.ats.ucla.edu/stat/spss/whatstat/whatstat.htm>

---

*mtcarz*

*mtcarz*

---

**Description**

Copy of mtcars data set with modified variable types

**Usage**

*mtcarz*

**Format**

An object of class `data.frame` with 32 rows and 11 columns.

# Index

## \*Topic **datasets**

- hsb, 32
  - mtcarz, 33
- Binomial, 3
- Chisquare, 4
- descriptr, 3
- descriptr-package (descriptr), 3
- dist\_binom\_perc (dist\_binom\_plot), 3
- dist\_binom\_plot, 3
- dist\_binom\_prob (dist\_binom\_plot), 3
- dist\_chi\_perc (dist\_chi\_plot), 4
- dist\_chi\_plot, 4
- dist\_chi\_prob (dist\_chi\_plot), 4
- dist\_f\_perc (dist\_f\_plot), 5
- dist\_f\_plot, 5
- dist\_f\_prob (dist\_f\_plot), 5
- dist\_norm\_perc (dist\_norm\_plot), 6
- dist\_norm\_plot, 6
- dist\_norm\_prob (dist\_norm\_plot), 6
- dist\_t, 7
- dist\_t\_perc (dist\_t), 7
- dist\_t\_plot (dist\_t), 7
- dist\_t\_prob (dist\_t), 7
- ds\_auto\_cross\_table  
(ds\_auto\_freq\_table), 8
- ds\_auto\_freq\_table, 8
- ds\_auto\_group\_summary, 9
- ds\_auto\_summary\_stats, 10
- ds\_cross\_table, 10, 13, 30
- ds\_css, 11
- ds\_cvar, 12
- ds\_extreme\_obs, 12
- ds\_freq\_cont (ds\_freq\_table), 13
- ds\_freq\_table, 13, 30
- ds\_gmean, 14, 16
- ds\_group\_summary, 14, 16
- ds\_group\_summary\_interact, 15
- ds\_hmean, 14, 16
- ds\_kurtosis, 17
- ds\_launch\_shiny\_app, 18
- ds\_mdev, 18
- ds\_measures\_location, 19
- ds\_measures\_symmetry, 20
- ds\_measures\_variation, 20
- ds\_mode, 21
- ds\_multi\_stats (ds\_tidy\_stats), 31
- ds\_oway\_tables (ds\_auto\_freq\_table), 8
- ds\_percentiles, 21
- ds\_plot\_bar, 22
- ds\_plot\_bar\_grouped, 23
- ds\_plot\_bar\_stacked, 23
- ds\_plot\_box\_group, 24
- ds\_plot\_box\_single, 24
- ds\_plot\_density, 25
- ds\_plot\_histogram, 25
- ds\_plot\_scatter, 26
- ds\_range, 26
- ds\_rindex, 27
- ds\_screener, 28
- ds\_skewness, 29
- ds\_std\_error, 30
- ds\_summary\_stats, 15, 30
- ds\_tailobs, 31
- ds\_tidy\_stats, 31
- ds\_tway\_tables (ds\_auto\_freq\_table), 8
- ds\_twoway\_table (ds\_cross\_table), 10
- FDist, 6
- hsb, 32
- mad, 19
- mean, 14, 16, 21
- median, 21
- mtcarz, 33
- Normal, 7

`plot.ds_cross_table(ds_cross_table)`, [10](#)

`plot.ds_freq_table(ds_freq_table)`, [13](#)

`plot.ds_group_summary`

`(ds_group_summary)`, [14](#)

`plot.ds_screener(ds_screener)`, [28](#)

`range`, [27](#)

`summary`, [30](#)

`TDist`, [8](#)

`top_n`, [31](#)