Package ‘descstat’

October 13, 2022

Version 0.1-2

Date 2021-02-17

Title Tools for Descriptive Statistics

Depends R (>= 4.0.0)

Imports rlang, purrr, dplyr, tidyr, tibble, tidyselect, forcats, cli, magrittr

Suggests knitr, ggplot2, rmarkdown

Description A toolbox for descriptive statistics, based on the computation of frequency and contingency tables. Several statistical functions and plot methods are provided to describe univariate or bivariate distributions of factors, integer series and numerical series either provided as individual values or as bins.

Encoding UTF-8

License GPL (>= 2)

URL `https://www.r-project.org`

VignetteBuilder knitr

RoxygenNote 7.1.1

LazyData true

NeedsCompilation no

Author Yves Croissant [aut, cre]

Maintainer Yves Croissant `<yves.croissant@univ-reunion.fr>`

Repository CRAN

Date/Publication 2021-02-17 16:40:02 UTC

R topics documented:

descstat-package 2
bin 3
bivariate 5
cont_table 7
employment 8

freq_table 9
income 11
padova 11
pre_plot 12
print_method 14
rgp 15
univariate 16
wages 19

Index **20**

`descstat-package` *descstat: a toolbox for descriptive statistics*

Description

Descriptive statistics consist on presenting the distribution of series for a sample in tables (frequency table for one series, contingency tables for two series), ploting this distribution and computing some statistics that summarise it. **descstat** provides a complete toolbox to perform this tasks. It has been writen using some packages of the tidyverse (especially **dplyr**, **tidyr** and **purrr**) and its usage follow the tidyverse conventions, especially the selection of series using their unquoted names and the use of the pipe operator and of tibbles.

The bin class

In a frequency (or contingency table), continuous numerical series are presented as bins. Moreover, for some surveys, the individual values are not known, but only the fact that these values belongs to a bin. Therefore, it is crucial to be able to work easily with bins, ie:

- creating bins from numerical values, which is performed by the `base::cut` function which turns a numerical series to a bin,
- coercing bins to numerical values, eg getting from the `[10,20)` bin the lower bound (10), the upper bound (20), the center (15) or whatever other value of the bin,
- reducing the number of bins by merging some of them (for example `[0,10)`, `[10, 20)`, `[20,30)`, `[30,Inf)` to `[0,20)`, `[20,Inf)`

these latter two tasks are performed using the new `bin` class provided by this package and the accompanying `as_numeric` function for the coercion to numeric and the `cut` method for bins merging. Especially, coercing bins to their center values is the basis of the computation of descripting statistics for bins.

Frequency and contingency tables

The `freq_table` and `cont_table` are based on the `dplyr::count` function but offer a much richer interface and performs easily usual operations which are tedious to obtain with `dplyr::count` or `base::table` functions. This includes:

- adding a total,

- for frequency tables, computing other kind of frequencies than the counts, for example relative frequencies, percentage, cummulative frequencies, etc.,
- for contingency tables, computing easily the joint, marginal and conditional distributions,
- printing easily the contingency table as a double entry table.

Plotting the distribution

A `pre_plot` function is provided to put the tibble in form in order to use classic plots for univariate or bivariate distributions. This includes histogram, frequency plot, pie chart, cummulative plot and Lorenz curve. The final plot can then be obtained using some geoms of **ggplot2**.

Descriptive statistics

A full set of statistical functions (of central tendency, dispersion, shape, concentration and covariation) are provided and can be applied directly on objects of class `freq_table` or `cont_table`. Some of them are methods of generics defined by the `base` or `stats` package, some other are defined as methods for generics function provided by the **descstat** function when the corresponding **R** function is not generic. For example,

- `mean` is generic, so that we wrote a `mean.freq_table` method to compute directly the mean of a series from a frequency table.
- `var` is not generic, so that we provide the `variance` generic and a method for `freq_table` objects.

bin *Bin series*

Description

A new class called `bin` is provided, along with different functions which enable to deal easily with bins, ie creating `bin` objects (`as_bin`) coercing bins to numerical values (`as_numeric`), merging bins (`cut`) and checking than an object is a bin (`is_bin`).

Usage

```
as_bin(x)

is_bin(x)

as_numeric(x, pos = 0, xfirst = NULL, xlast = NULL, wlast = NULL)

## S3 method for class 'bin'
cut(x, breaks = NULL, ...)

## S3 method for class 'character'
cut(x, breaks = NULL, ...)
```

```
## S3 method for class 'factor'
cut(x, breaks = NULL, ...)

## S3 method for class 'character'
extract(data, ..., .name_repair = "check_unique")

## S3 method for class 'factor'
extract(data, ..., .name_repair = "check_unique")
```

Arguments

`x`	a character or a factor: the first and last characters should be any of `[`, `(`, `]`, `)` and the other characters should be interpreted as two numerical values separated by a comma,
`pos`	a numeric between 0 and 1, 0 for the lower bond, 1 for the upper bond, 0.5 for the center of the class (or any other value between 0 and 1), which indicates to `as_numeric` how the bins should be coerced to numerical values,
`xfirst, xlast`	the center of the first (last) class, if one wants to specify something different from the average of the lower and the upper bonds,
`wlast`	in the case where the upper bond is infinite and `xlast` is not provided, the width of the last class is set to the one of the before last class. If `wlast` is provided, it is set to the width of the before last class times `wlast`,
`breaks`	a numerical vector of breaks which should be a subset of the initial set of breaks. If only one break is provided, all the bins with greater values are merged,
`...`	see `base::cut()` for the `cut` method and `tidyr::extract()` for the `extract` method,
`data`	a character or a factor containing bins,
`.name_repair`	see `tidyr::extract()`.

Details

- `extract` methods for characters and factors are provided which split the character strings in a four tibble columns: the open bracket, the lower bound, the upper bound and the closing bracket.
- `as_bin` takes as argument a character or a factor that represents a bin, check the consistency of the string and return a bin object with levels in the correct order and NAs when the strings are malformed,
- the default `cut` method takes a numerical series as argument and returns a factor containing bins according to a `break` vector; for the bin's method, the break should be a subset of the original set of breaks and a bin with fewer levels results,
- `as_numeric` converts a bin to a value of the underlying variable defined by its relative position (from 0 lower bound to 1 upper bound in the bin),
- `is_bin` check if the argument is a bin.

Value

`as_bin` returns a `bin` object, `is_bin` a logical, the `extract` method a tibble, `as_numeric` a numeric and the `cut` method a `bin` object with fewer levels.

Author(s)

Yves Croissant

Examples

```
# create a factor containing bins using cut on a numeric
z <- c(1, 5, 10, 12, 4, 9, 8)
bin1 <- cut(z, breaks = c(1, 8, 12, Inf), right = FALSE)
# extract the elements of the levels in a tibble
extract(bin1)
# coerce to a bin object
bin2 <- as_bin(bin1)
# coerce to a numeric using the center of the bins
as_numeric(bin2, pos = 0.5)
# special values for the center of the first and of the last bin
as_numeric(bin2, pos = 0.5, xfirst = 5, xlast = 16)
# same, but indicating that the width of the last class should be
# twice the one of the before last
as_numeric(bin2, pos = 0.5, xfirst = 5, wlast = 2)
# merge in order to get only two bins
cut(bin2, breaks = c(1, 12))
# if length of breaks is 1, this is the value for which all the bins
# containing greater values are merged
cut(bin2, breaks = 8)
# check that bin1 and bin2 are objects of class bin
is_bin(bin1)
is_bin(bin2)
```

`bivariate` *Functions to compute statistics on bivariate distributions*

Description

These functions are intended to compute from a `cont_table` objects covariation statistics, ie the covariance, the correlation coefficient, variance decomposition and regression line.

Usage

```
covariance(data, ...)

correlation(data, ...)

## S3 method for class 'cont_table'
covariance(data, ...)
```

```
## S3 method for class 'cont_table'
correlation(data, ...)

## S3 method for class 'cont_table'
anova(object, x, ...)

## S3 method for class 'anova.cont_table'
summary(object, ...)

regline(formula, data)
```

Arguments

`data, object`	a `cont_table` object,
`...`	further arguments.
`x`	the series for which the analyse of variance should be computed,
`formula`	symbolic description of the model,

Value

a numeric or a tibble

Author(s)

Yves Croissant

Examples

```
# the covariance and the linear correlation coefficient are
# computed using only the `cont_table`
# First reduce the number of bins
wages2 <- wages %>%
          dplyr::mutate(size = cut(as_bin(size), c(20, 50, 100)),
                        wage = cut(as_bin(wage), c(10, 30, 50)))
wages2 %>% cont_table(wage, size) %>% covariance
wages2 %>% cont_table(wage, size) %>% correlation
# For the analyse of variance, one of the two series should be
# indicated
wages2 %>% cont_table(wage, size) %>% anova(wage)
wages2 %>% cont_table(wage, size) %>% anova(wage) %>% summary
# For the regression line, a formula should be provided
wages2 %>% cont_table(wage, size) %>% regline(formula = wage ~ size)
```

`cont_table` *Contingency table*

Description

A contingency table returns the counts of all the combinations of the modalities of two series in a table for which every modality of the first series is a row and every modality of the second series is a column. The `joint`, `marginal` and `conditional` functions compute these three distributions from the contingency table (by indicating one series for the last two).

Usage

```
cont_table(
  data,
  x1,
  x2,
  weights = NULL,
  freq = NULL,
  total = FALSE,
  xfirst1 = NULL,
  xlast1 = NULL,
  wlast1 = NULL,
  xfirst2 = NULL,
  xlast2 = NULL,
  wlast2 = NULL
)

joint(data)

conditional(data, x = NULL)

marginal(data, x = NULL, f = "f", vals = NULL)
```

Arguments

`data`	a tibble,
`x1, x2`	the two series used the construct the contingency table, the distinct values of the first and the second will respectively be the rows and the columns of the contingency table,
`weights`	a series containing the weights that should be used to mimic the population,
`freq`	the frequencies (in the case where data is already contingency table),
`total`	if `TRUE`, a total is added to the table,
`xfirst1, xfirst2, xlast1, xlast2, wlast1, wlast2`	see `as_numeric()`,
`x`	the series on which the operation should be computed,

`f`	see `freq_table()`,
`vals`	see `freq_table()`,

Details

`cont_table` actually returns a tibble in "long format", as the `dplyr::count` table does. As the returned object is of class `cont_table`, this is the `format` and `print` methods that turns the tibble in a wide format before printing.

The `conditional` and `joint` functions return a `cont_table` object, as the `marginal` function returns a `freq_table` object.

Value

a tibble

Author(s)

Yves Croissant

Examples

```
library("dplyr")
# get a contingency table containing education and sex
cont_table(employment, education, sex)
# instead of counts, sum the weights
cont_table(employment, education, sex, weights = weights)
# get the joint distribution and the conditional and marginal
# distribution of sex
cont_table(employment, education, sex) %>% joint
cont_table(employment, education, sex) %>% marginal(sex)
cont_table(employment, education, sex) %>% conditional(sex)
```

`employment` *French employment survey*

Description

The employment survey gives information about characteristics of a sample of individuals (employed/unemployed, part/full time job, education, etc.).

Format

a tibble containing

- activity : a factor with levels `occupied`, `unemployed` and `inactive`,
- time : job time a factor with levels `part`, `full` and `unknown`,
- education : level of education,

- age : age in years,
- sex : one of `male` or `female`,
- household : kind of household, `single`, `monop` (mono-parental family), `couple` (couple without children), `family` (couple with families) and `other`,
- weights : weights to mimic the population.

Source

Employment survey 2018, INSEE's website.

`freq_table` *Frequency table*

Description

Compute the frequency table of a categorical or a numerical series.

Usage

```
freq_table(
  data,
  x,
  f = "n",
  vals = NULL,
  weights = NULL,
  total = FALSE,
  max = NULL,
  breaks = NULL,
  right = NULL,
  xfirst = NULL,
  xlast = NULL,
  wlast = NULL,
  freq = NULL,
  mass = NULL,
  center = NULL
)
```

Arguments

`data`	a tibble,
`x`	a categorical or numerical series,
`f`	a string containing `n` for counts, `f` for relative frequencies, `p` for percentages and `m` for mass frequencies. Cumulative series are obtained using the same letters in upper caps,

`vals`	a character containing letters indicating the values of the variable that should be returned; `x` for the center of the class, `l` and `u` for the lower and upper limit of the class, `a` for the range,
`weights`	a series that contain the weights that enable the sample to mimic the population,
`total`	a logical indicating whether the total should be returned,
`max`	if the series is a discrete numerical value, this argument indicates that all the values greater than `max` should be merged in the same modality,
`breaks`	a numerical vector of class limits,
`right`	a logical indicating whether classes should be closed (`right = TRUE`) or open (`right = FALSE`) on the right,
`xfirst, xlast, wlast`	see `as_numeric()`,
`freq`	a series that contains the frequencies (only relevant if `data` is already a frequency table),
`mass`	a series that contains the masses of the variable (only relevant if `data` is already a frequency table),
`center`	a series that contains the center of the class of the variable (only relevant if `data` is already a frequency table).

Value

a tibble containing the specified values of `vals` and `f`.

Author(s)

Yves Croissant

Examples

```
# in table padova, price is a numeric variable, a vector of breaks should be provided
library("dplyr")
padova %>% freq_table(price,
                      breaks = c(50, 100, 150, 200, 250, 300, 350, 400),
                      right = TRUE)
# return relative frequencies and densities, and the center value
# of the series and the width of the bin
padova %>% freq_table(price,
                      breaks = c(50, 100, 150, 200, 250, 300, 350, 400),
                      right = TRUE, f = "fd", vals = "xa")
# in table wages, wage is a factor that represents the classes
wages %>% freq_table(wage, "d")
# a breaks argument is provided to reduce the number of classes
wages %>% freq_table(wage, breaks = c(10, 20, 30, 40, 50))
# a total argument add a total to the frequency table
wages %>% freq_table(wage, breaks = c(10, 20, 30, 40, 50), total = TRUE)
# income is already a frequency table, the freq argument
# is mandatory
income %>% freq_table(inc_class, freq = number)
```

```
# the mass argument can be indicated if on column contains the
# mass of the series in each bin. In this case, the center of the
# class are exactly the mean of the series in each bin
income %>% freq_table(inc_class, freq = number, mass = tot_inc)
# rgp contains a children series which indicates the number of
# children of the households
rgp %>% freq_table(children)
# a max argument can be indicated to merge the unusual high
# values of number of childre
rgp %>% freq_table(children, max = 4)
# employment is a non random survey, there is a weights series
# that can be used to compute the frequency table according to the
# sum of weights and not to counts
employment %>% freq_table(education)
employment %>% freq_table(education, weights = weights)
```

`income` *Income of French households*

Description

Bins of income classes, number of households and mass of income.

Format

a tibble containing :

- bin: bin of income,
- number: number of households in the bin,
- income: mass of income in the bin.

Source

Impot sur le revenu par commune (IRCOM) DGI's website.

`padova` *Housing prices in Padova*

Description

This data set documents characteristics (including the prices) of a sample of housings in Padova.

Format

a tibble containing

- zone : one of the 12 zones of Padova,
- condition : `new` for new housings, `ordinary` or `good` for old ones,
- house : dummy for houses,
- floor : floor,
- rooms : number of rooms,
- bathrooms : number of bathrooms,
- parking : dummy for parkings,
- energy : energy cathegory for the house (A for the best, G for the worst),
- area : area of the house in square meters,
- price : price of the house in thousands of euros.

Source

Data in Brief's website.

References

Bonifaci P, Copiello S (2015). "Real estate market and building energy performance: Data for a mass appraisal approach." *Data in Brief*, *5*, 1060-1065. ISSN 2352-3409.

`pre_plot` *Put a tibble in form to plot*

Description

Convert a tibble built using `freq_table` or `cont_table` in a shape that makes it easy to plot.

Usage

```
pre_plot(data, f = NULL, plot = NULL, ...)

## S3 method for class 'freq_table'
pre_plot(
  data,
  f = NULL,
  plot = c("histogram", "freqpoly", "lorenz", "stacked", "cumulative"),
  ...
)

## S3 method for class 'cont_table'
pre_plot(data, ...)
```

Arguments

`data`	a tibble returned by the `freq_table` or the `cont_table` function, which should contain the center of the classes (`x`) and at least one measure of the frequencies or densities (one of `f`, `n`, `p`, `d`),
`f`	mandatory argument if the tibble contains more than one frequency or density,
`plot`	for object of class `freq_table` one of `histogram`, `freqpoly`, `stacked`, `cumulative` and `lorenz` (see the details section),
`...`	further arguments.

Details

The `pre_plot` function returns a tibble containing:

- if `plot` = `histogram`, `x`, `y` that should be passed to `geom_polygon`,
- if `plot` = `freqpoly` `x` and `y` that should be passed to `geom_line`,
- if `plot` = `stacked` `x` and `ypos` that should be passed respectively to `geom_col` and to `geom_text` to draw labels on the right position,
- if `plot` = `cumulative` `x`, `y`, `xend` and `yend` that should be passed to `geom_segment`,
- if `plot` = `lorenz` for the Lorenz curve, `F` and `M` for the coordinates of the polygons under the Lorenz curve, `pts` is logical which the defines the subset of points that belongs to the Lorenz curve.

Value

a tibble

Author(s)

Yves Croissant

Examples

```
library("dplyr")
library("ggplot2")
pad <- padova %>%
       freq_table(price, breaks = c(100, 200, 300, 400, 500, 1000),
       right = TRUE, f = "Npd")
pad %>% pre_plot(f = "d") %>% ggplot() + geom_polygon(aes(x, y))
pad %>% pre_plot(f = "d", plot = "freqpoly") %>%
ggplot() + geom_line(aes(x, y))
## A pie chart
wages %>% freq_table(sector, "p", total = FALSE) %>%
  pre_plot("p", plot = "stacked") %>% ggplot(aes(x = 2, y = p, fill = sector)) +
  geom_col() + geom_text(aes(y = ypos, label = sector)) +
  coord_polar(theta = "y") + theme_void() + guides(fill = FALSE)
```

print_method *Print methods for bin, freq_table and cont_table objects*

Description

freq_table and cont_table are tibbles with specific format and print methods for pretty printing. A pre_print generic is provided with specific methods to put in form freq_table and cont_table objects.

Usage

```
pre_print(x, ...)

## S3 method for class 'freq_table'
pre_print(x, ...)

## S3 method for class 'cont_table'
pre_print(x, ..., row_name = TRUE, total_name = "Total")

## S3 method for class 'freq_table'
format(x, ..., n = NULL, width = NULL, n_extra = NULL)

## S3 method for class 'cont_table'
format(
  x,
  ...,
  n = NULL,
  width = NULL,
  n_extra = NULL,
  row_name = TRUE,
  total_name = "Total"
)

## S3 method for class 'cont_table'
print(
  x,
  ...,
  n = NULL,
  width = NULL,
  n_extra = NULL,
  row_name = TRUE,
  total_name = "Total"
)

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

Arguments

`x`	a `bin`, a `freq_table` or a `cont_table` object,
`...`	further arguments,
`row_name`	a logical that indicates whether the first column in the two-ways contingency table, that contains the levels of the first series, should be named,
`total_name`	the name of the line (and of the column for `cont_table`) that contains the total (default is `"Total"`),
`n, width, n_extra`	see tibble::formatting and tibble::formatting.

Value

a tibble, for the `cont_table` it is a tibble in wide format as the `cont_table` object is in long format.

`rgp` *Extract of the French census*

Description

This extract of the French census gives information about a sample of French households.

Format

a tibble containing :

- cars : number of cars,
- rooms : number of rooms of the housing,
- children : number of children,
- type : type of household ; `couple` or `monop` (for mono-parental families),

Source

INSEE's website.

`univariate` *Functions to compute statistics on univariate distributions*

Description

descstat provide functions to compute statistics on an univariate distribution. This includes central tendency, dispersion, shape and concentration.

Usage

```
variance(x, ...)

gmean(x, r = 1, ...)

gini(x, ...)

stdev(x, ...)

madev(x, ...)

modval(x, ...)

medial(x, ...)

kurtosis(x, ...)

skewness(x, ...)

## Default S3 method:
variance(x, w = NULL, ...)

## Default S3 method:
gmean(x, r = 1, ...)

## Default S3 method:
stdev(x, w = NULL, ...)

## Default S3 method:
madev(x, w = NULL, center = c("median", "mean"), ...)

## Default S3 method:
skewness(x, ...)

## Default S3 method:
kurtosis(x, ...)

## S3 method for class 'freq_table'
```

```
mean(x, ...)

## S3 method for class 'freq_table'
gmean(x, r = 1, ...)

## S3 method for class 'freq_table'
variance(x, ...)

## S3 method for class 'freq_table'
stdev(x, ...)

## S3 method for class 'freq_table'
skewness(x, ...)

## S3 method for class 'freq_table'
kurtosis(x, ...)

## S3 method for class 'freq_table'
madev(x, center = c("median", "mean"), ...)

## S3 method for class 'freq_table'
modval(x, ...)

## S3 method for class 'freq_table'
quantile(x, y = c("value", "mass"), probs = c(0.25, 0.5, 0.75), ...)

## S3 method for class 'freq_table'
median(x, ..., y = c("value", "mass"))

## S3 method for class 'freq_table'
medial(x, ...)

## S3 method for class 'freq_table'
gini(x, ...)

## S3 method for class 'cont_table'
modval(x, ...)

## S3 method for class 'cont_table'
gini(x, ...)

## S3 method for class 'cont_table'
skewness(x, ...)

## S3 method for class 'cont_table'
kurtosis(x, ...)

## S3 method for class 'cont_table'
```

```
madev(x, center = c("median", "mean"), ...)

## S3 method for class 'cont_table'
mean(x, ...)

## S3 method for class 'cont_table'
variance(x, ...)

## S3 method for class 'cont_table'
stdev(x, ...)
```

Arguments

`x`	a series or a `freq_table` or a `cont_table` object,
`...`	further arguments,
`r`	the order of the mean for the `gmean` function,
`w`	a vector of weights,
`center`	the center value used to compute the mean absolute deviations, one of `"median"` or `"mean"`,
`y`	for the quantile method, one of `"value"` or `"mass"`,
`probs`	the probabilities for which the quantiles have to be computed.

Details

The following functions are provided:

- central tendency: `mean`, `median`, `medial`, `modval` (for the mode),
- dispersion: `variance`, `stdev`, `maddev` (for mean absolute deviation) and quantile,
- shape: `skewness` and `kurtosis`,
- concentration: `gini`.

When a generic function exists in base **R** (or in the `stats` package), methods are provided for `freq_table` or `cont_table`, this is a case for `mean`, `median` and `quantile`. When a function exists, but is not generic, we provide a generic and relevant methods using different names (`stdev`, `variance` and `madev` instead respectively of `sd`, `var` and `mad`). Finally some function don't exist in base **R** and recommended packages, we therefore provide a `modval` function to compute the mode, `gini` for the Gini concentration index, `skewness` and `kurtosis` for Fisher's shape statistics and `gmean` for generalized means (which include the geometric, the quadratic and the harmonic means).

`madev` has a center argument which indicates whether the deviations should be computed respective to the mean or to the median.

`gmean` has a `r` argument: values of -1, 0, 1 and 2 lead respectively to the harmonic, geometric, arithmetic and quadratic means.

Value

a numeric or a tibble.

Author(s)

Yves Croissant

Examples

```
library("dplyr")
z <- wages %>% freq_table(wage)
z %>% median
# the medial is the 0.5 quantile of the mass of the distribution
z %>% medial
# the modval function returns the mode, it is a one line tibble
z %>% modval
z %>% quantile(probs = c(0.25, 0.5, 0.75))
# quantiles can compute for the frequency (the default) or the mass
# of the series
z %>% quantile(y = "mass", probs = c(0.25, 0.5, 0.75))
# univariate statistics can be computed on the joint, marginal or
# conditional distributions for cont_table objects
wages %>% cont_table(wage, size) %>% joint
wages %>% cont_table(wage, size) %>% marginal(size) %>% mean
wages %>% cont_table(wage, size) %>% conditional(size) %>% mean
```

`wages` *DADS survey*

Description

The DADS survey (Declaration Annuelle des Données Sociales) provides characteristics of wage earners (wages in class, number of working hours, etc.).

Format

a tibble containing

- sector : activity sector, `industry`, `building`, `business`, `services` and `administration`,
- age : the age in years,
- hours : annual number of hours worked,
- sex : sex of the wage earner, `male` or `female`,
- wage : class of yearly wages, in thousands of euros,
- size : class of working force size of the firm.

Source

DADS survey 2015, INSEE's website.

Index

∗ **bivariate**
  bivariate, 5
∗ **classes**
  bin, 3
∗ **datasets**
  employment, 8
  income, 11
  padova, 11
  rgp, 15
  wages, 19
∗ **dplot**
  pre_plot, 12
∗ **manip**
  cont_table, 7
  freq_table, 9
∗ **package**
  descstat-package, 2
∗ **print**
  print_method, 14
∗ **univar**
  univariate, 16

anova.cont_table *(bivariate)*, 5
as_bin *(bin)*, 3
as_numeric *(bin)*, 3
as_numeric(), *7*, *10*

base::cut(), *4*
bin, 3
bivariate, 5

conditional *(cont_table)*, 7
cont_table, 7
correlation *(bivariate)*, 5
covariance *(bivariate)*, 5
cut.bin *(bin)*, 3
cut.character *(bin)*, 3
cut.factor *(bin)*, 3

descstat-package, 2

employment, 8
extract.character *(bin)*, 3
extract.factor *(bin)*, 3

format.cont_table *(print_method)*, 14
format.freq_table *(print_method)*, 14
freq_table, 9
freq_table(), *8*

gini *(univariate)*, 16
gmean *(univariate)*, 16

income, 11
is_bin *(bin)*, 3

joint *(cont_table)*, 7

kurtosis *(univariate)*, 16

madev *(univariate)*, 16
marginal *(cont_table)*, 7
mean.cont_table *(univariate)*, 16
mean.freq_table *(univariate)*, 16
medial *(univariate)*, 16
median.freq_table *(univariate)*, 16
modval *(univariate)*, 16

padova, 11
pre_plot, 12
pre_print *(print_method)*, 14
print.bin *(print_method)*, 14
print.cont_table *(print_method)*, 14
print_method, 14

quantile.freq_table *(univariate)*, 16

regline *(bivariate)*, 5
rgp, 15

skewness *(univariate)*, 16
stdev *(univariate)*, 16

summary.anova.cont_table *(*bivariate*)*, 5

tibble::formatting, *15*
tidyr::extract(), *4*

univariate, 16

variance *(*univariate*)*, 16

wages, 19