

Package ‘radiant’

October 25, 2015

Title Business Analytics using R and Shiny

Version 0.3.56

Date 2015-10-24

Description A platform-independent browser-based interface for business analytics in R, based on the Shiny package.

Depends R (>= 3.2.0),
magrittr (>= 1.5),
ggplot2 (>= 1.0.0),
lubridate (>= 1.3.3),
tidyr (>= 0.3.1),
dplyr (>= 0.4.3)

Imports DiagrammeR(>= 0.7),
car (>= 2.0.22),
MASS (>= 7.3),
gridExtra (>= 2.0.0),
AlgDesign (>= 1.1.7.3),
psych (>= 1.4.8.11),
GPArotation (>= 2014.11.1),
wordcloud (>= 2.5),
markdown (>= 0.7.4),
knitr (>= 1.8),
ggdendro (>= 0.1.17),
broom (>= 0.3.7),
pryr (>= 0.1),
shiny (>= 0.12.2),
shinyAce (>= 0.2.1),
DT (>= 0.1.39),
MathJaxR (>= 0.11),
readr (>= 0.1.1),
data.tree(>= 0.2.1),
yaml(>= 2.1.13),
scales(>= 0.2.5),
curl(>= 0.9.1),
stringr (>= 1.0)

Suggests rmarkdown (>= 0.4.2),
devtools (>= 1.8.0),
testthat (>= 0.10.0),
covr (>= 1.2.0)

URL <https://github.com/vnijs/radiant>, <http://vnijs.github.io/radiant/>

BugReports <https://github.com/vnijs/radiant/issues>

License AGPL-3 | file LICENSE

LazyData true

R topics documented:

| | |
|-------------------|----|
| as_character | 6 |
| as_distance | 6 |
| as_dmy | 7 |
| as_dmy_hm | 7 |
| as_dmy_hms | 8 |
| as_duration | 8 |
| as_factor | 9 |
| as_hm | 9 |
| as_hms | 10 |
| as_integer | 10 |
| as_mdy | 11 |
| as_mdy_hm | 11 |
| as_mdy_hms | 12 |
| as_numeric | 12 |
| as_ymd | 13 |
| as_ymd_hm | 13 |
| as_ymd_hms | 14 |
| avengers | 14 |
| center | 15 |
| changedata | 15 |
| city | 16 |
| ci_label | 16 |
| ci_perc | 17 |
| clean_loadings | 17 |
| combinedata | 18 |
| compare_means | 19 |
| compare_props | 20 |
| computer | 21 |
| conjoint | 21 |
| conjoint_profiles | 22 |
| copy_all | 23 |
| copy_from | 23 |
| correlation | 24 |
| cross_tabs | 25 |
| cv | 26 |
| decile_split | 26 |
| diamonds | 27 |
| does_vary | 27 |
| dtree | 28 |
| dtree_parser | 28 |
| explore | 29 |
| factorizer | 30 |
| ff_design | 30 |

| | |
|------------------------------|----|
| filterdata | 31 |
| flip | 31 |
| full_factor | 32 |
| getclass | 33 |
| getdata | 33 |
| getsummary | 34 |
| glm_reg | 34 |
| hier_clus | 35 |
| inverse | 36 |
| is_empty | 37 |
| is_string | 37 |
| items | 38 |
| kmeans_clus | 38 |
| kurtosi | 39 |
| launcher | 40 |
| lin_launcher | 40 |
| loadcsv | 41 |
| loadcsv_url | 42 |
| loadrda | 42 |
| loadrda_url | 43 |
| mac_launcher | 43 |
| make_dt | 44 |
| make_expl | 44 |
| make_funs | 45 |
| make_train | 46 |
| max_rm | 46 |
| mds | 47 |
| mean_rm | 48 |
| median_rm | 48 |
| median_split | 49 |
| min_rm | 49 |
| mode_rm | 50 |
| mp3 | 50 |
| mutate_each | 51 |
| newspaper | 51 |
| normalize | 52 |
| n_missing | 52 |
| p05 | 53 |
| p25 | 53 |
| p75 | 54 |
| p95 | 54 |
| pivotr | 55 |
| plot.compare_means | 56 |
| plot.compare_props | 56 |
| plot.conjoint | 57 |
| plot.correlation_ | 58 |
| plot.cross_tabs | 58 |
| plot.dtree | 59 |
| plot.full_factor | 60 |
| plot.glm_predict | 61 |
| plot.glm_reg | 62 |
| plot.hier_clus | 63 |

| | |
|--------------------|----|
| plot.kmeans_clus | 64 |
| plot.mds | 64 |
| plot.pivotr | 65 |
| plot.pmap | 66 |
| plot.pre_factor | 67 |
| plot.prob_binom | 67 |
| plot.prob_chisq | 68 |
| plot.prob_disc | 68 |
| plot.prob_fdist | 69 |
| plot.prob_norm | 69 |
| plot.prob_tdist | 70 |
| plot.prob_unif | 70 |
| plot.regression | 71 |
| plot.reg_predict | 72 |
| plot.repeater | 73 |
| plot.simulator | 73 |
| plot.single_mean | 74 |
| plot.single_prop | 75 |
| pmap | 75 |
| predict.glm_reg | 76 |
| predict.regression | 77 |
| pre_factor | 78 |
| print.gtable | 79 |
| prob_binom | 79 |
| prob_chisq | 80 |
| prob_disc | 80 |
| prob_fdist | 81 |
| prob_norm | 81 |
| prob_tdist | 82 |
| prob_unif | 82 |
| publishers | 83 |
| radiant | 83 |
| recode | 84 |
| regression | 84 |
| repeater | 85 |
| rndnames | 85 |
| sample_size | 86 |
| sampling | 87 |
| save_factors | 87 |
| save_membership | 88 |
| sdp_rm | 89 |
| sd_rm | 89 |
| serr | 90 |
| set_class | 90 |
| shopping | 91 |
| show_duplicated | 91 |
| sig_stars | 92 |
| simulator | 92 |
| single_mean | 93 |
| single_prop | 94 |
| skew | 95 |
| square | 95 |

| | |
|---------------------------|-----|
| sshh | 96 |
| sshhr | 96 |
| standardize | 97 |
| state_init | 97 |
| state_multiple | 98 |
| state_single | 99 |
| store_glm | 100 |
| store_reg | 100 |
| summary.compare_means | 101 |
| summary.compare_props | 102 |
| summary.conjoint | 102 |
| summary.conjoint_profiles | 103 |
| summary.correlation_ | 104 |
| summary.cross_tabs | 104 |
| summary.dtree | 105 |
| summary.explore | 106 |
| summary.full_factor | 106 |
| summary.glm_reg | 107 |
| summary.hier_clus | 108 |
| summary.kmeans_clus | 109 |
| summary.mds | 109 |
| summary.pivotr | 110 |
| summary.pmap | 111 |
| summary.pre_factor | 112 |
| summary.prob_binom | 112 |
| summary.prob_chisq | 113 |
| summary.prob_disc | 113 |
| summary.prob_fdist | 114 |
| summary.prob_norm | 114 |
| summary.prob_tdist | 115 |
| summary.prob_unif | 115 |
| summary.regression | 116 |
| summary.repeater | 117 |
| summary.sample_size | 117 |
| summary.sampling | 118 |
| summary.simulator | 118 |
| summary.single_mean | 119 |
| summary.single_prop | 120 |
| sum_rm | 120 |
| superheroes | 121 |
| test_specs | 121 |
| the_table | 122 |
| titanic | 122 |
| titanic_pred | 123 |
| toothpaste | 123 |
| varp_rm | 124 |
| var_check | 124 |
| var_rm | 125 |
| viewdata | 125 |
| visualize | 126 |
| win_launcher | 127 |

| | |
|--------------|---------------------------------|
| as_character | <i>Wrapper for as.character</i> |
|--------------|---------------------------------|

Description

Wrapper for as.character

Usage

```
as_character(x)
```

Arguments

| | |
|---|--------------|
| x | Input vector |
|---|--------------|

| | |
|-------------|--|
| as_distance | <i>Distance in kilometers or miles between two locations based on lat-long Function based on http://www.movable-type.co.uk/scripts/latlong.html. Uses the haversine formula</i> |
|-------------|--|

Description

Distance in kilometers or miles between two locations based on lat-long Function based on <http://www.movable-type.co.uk/scripts/latlong.html>. Uses the haversine formula

Usage

```
as_distance(lat1, long1, lat2, long2, unit = "km", R = c(km = 6371, miles = 3959)[[unit]])
```

Arguments

| | |
|-------|---|
| lat1 | Latitude of location 1 |
| long1 | Longitude of location 1 |
| lat2 | Latitude of location 2 |
| long2 | Longitude of location 2 |
| unit | Measure kilometers ("km", default) or miles ("miles") |
| R | Radius of the earth |

Value

Distance bewteen two points

Examples

```
as_distance(32.8245525,-117.0951632, 40.7033127,-73.979681, unit = "km")
as_distance(32.8245525,-117.0951632, 40.7033127,-73.979681, unit = "miles")
```

`as_dmy`*Convert input in day-month-year format to date*

Description

Convert input in day-month-year format to date

Usage

```
as_dmy(x)
```

Arguments

`x` Input variable

Value

Date variable of class Date

Examples

```
as_dmy("1-2-2014")
```

`as_dmy_hm`*Convert input in day-month-year-hour-minute format to date-time*

Description

Convert input in day-month-year-hour-minute format to date-time

Usage

```
as_dmy_hm(x)
```

Arguments

`x` Input variable

Value

Date-time variable of class Date

Examples

```
as_dmy_hm("1-1-2014 12:15")
```

| | |
|------------|---|
| as_dmy_hms | <i>Convert input in day-month-year-hour-minute-second format to date-time</i> |
|------------|---|

Description

Convert input in day-month-year-hour-minute-second format to date-time

Usage

```
as_dmy_hms(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Date-time variable of class Date

Examples

```
as_dmy_hms("1-1-2014 12:15:01")
```

| | |
|-------------|--|
| as_duration | <i>Wrapper for lubridate's as.duration function. Result converted to numeric</i> |
|-------------|--|

Description

Wrapper for lubridate's as.duration function. Result converted to numeric

Usage

```
as_duration(x)
```

Arguments

| | |
|---|-----------------|
| x | Time difference |
|---|-----------------|

| | |
|-----------|------------------------------|
| as_factor | <i>Wrapper for as.factor</i> |
|-----------|------------------------------|

Description

Wrapper for as.factor

Usage

```
as_factor(x)
```

Arguments

| | |
|---|--------------|
| x | Input vector |
|---|--------------|

| | |
|-------|--|
| as_hm | <i>Convert input in hour-minute format to time</i> |
|-------|--|

Description

Convert input in hour-minute format to time

Usage

```
as_hm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Time variable of class Period

Examples

```
as_hm("12:45")  
## Not run:  
as_hm("12:45") %>% minute  
  
## End(Not run)
```

| | |
|--------|---|
| as_hms | <i>Convert input in hour-minute-second format to time</i> |
|--------|---|

Description

Convert input in hour-minute-second format to time

Usage

```
as_hms(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Time variable of class Period

Examples

```
as_hms("12:45:00")
## Not run:
as_hms("12:45:00") %>% hour
as_hms("12:45:00") %>% second

## End(Not run)
```

| | |
|------------|---|
| as_integer | <i>Convert variable to integer avoiding potential issues with factors</i> |
|------------|---|

Description

Convert variable to integer avoiding potential issues with factors

Usage

```
as_integer(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Integer

Examples

```
as_integer(rnorm(10))
as_integer(letters)
as_integer(5:10 %>% as.factor)
as.integer(5:10 %>% as.factor)
```

| | |
|--------|---|
| as_mdy | <i>Convert input in month-day-year format to date</i> |
|--------|---|

Description

Convert input in month-day-year format to date

Usage

```
as_mdy(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Details

Use as.character if x is a factor

Value

Date variable of class Date

Examples

```
as_mdy("2-1-2014")
## Not run:
as_mdy("2-1-2014") %>% month(label = TRUE)
as_mdy("2-1-2014") %>% week
as_mdy("2-1-2014") %>% wday(label = TRUE)

## End(Not run)
```

| | |
|-----------|--|
| as_mdy_hm | <i>Convert input in month-day-year-hour-minute format to date-time</i> |
|-----------|--|

Description

Convert input in month-day-year-hour-minute format to date-time

Usage

```
as_mdy_hm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Date-time variable of class Date

Examples

```
as_mdy_hm("1-1-2014 12:15")
```

```
as_mdy_hms
```

Convert input in month-day-year-hour-minute-second format to date-time

Description

Convert input in month-day-year-hour-minute-second format to date-time

Usage

```
as_mdy_hms(x)
```

Arguments

x Input variable

Value

Date-time variable of class Date

Examples

```
as_mdy_hms("1-1-2014 12:15:01")
```

```
as_numeric
```

Convert variable to numeric avoiding potential issues with factors

Description

Convert variable to numeric avoiding potential issues with factors

Usage

```
as_numeric(x)
```

Arguments

x Input variable

Value

Numeric

Examples

```
as_numeric(rnorm(10))
as_numeric(letters)
as_numeric(5:10 %>% as.factor)
as.numeric(5:10 %>% as.factor)
as_numeric(c("1", "2"))
```

| | |
|--------|---|
| as_ymd | <i>Convert input in year-month-day format to date</i> |
|--------|---|

Description

Convert input in year-month-day format to date

Usage

```
as_ymd(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Date variable of class Date

Examples

```
as_ymd("2013-1-1")
```

| | |
|-----------|--|
| as_ymd_hm | <i>Convert input in year-month-day-hour-minute format to date-time</i> |
|-----------|--|

Description

Convert input in year-month-day-hour-minute format to date-time

Usage

```
as_ymd_hm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Date-time variable of class Date

Examples

```
as_ymd_hm("2014-1-1 12:15")
```

| | |
|------------|---|
| as_ymd_hms | <i>Convert input in year-month-day-hour-minute-second format to date-time</i> |
|------------|---|

Description

Convert input in year-month-day-hour-minute-second format to date-time

Usage

```
as_ymd_hms(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Date-time variable of class Date

Examples

```
as_ymd_hms("2014-1-1 12:15:01")
## Not run:
as_ymd_hms("2014-1-1 12:15:01") %>% as.Date
as_ymd_hms("2014-1-1 12:15:01") %>% month
as_ymd_hms("2014-1-1 12:15:01") %>% hour

## End(Not run)
```

| | |
|----------|-----------------|
| avengers | <i>Avengers</i> |
|----------|-----------------|

Description

Avengers

Usage

```
data(avengers)
```

Format

A data frame with 7 rows and 4 variables

Details

List of avengers. The dataset is used to illustrate data merging / joining. Description provided in `attr(avengers,"description")`

| | |
|--------|---------------|
| center | <i>Center</i> |
|--------|---------------|

Description

Center

Usage

```
center(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

If x is a numeric variable return $x - \text{mean}(x)$

| | |
|------------|--------------------|
| changedata | <i>Change data</i> |
|------------|--------------------|

Description

Change data

Usage

```
changedata(dataset, vars = c(), var_names = names(vars))
```

Arguments

| | |
|-----------|--|
| dataset | Name of the dataframe to change |
| vars | New variables to add to the data.frame |
| var_names | Names for the new variables to add to the data.frame |

Value

None

Examples

```
r_data <- list()
r_data$dat <- data.frame(a = 1:20)
changedata("dat", 20:1, "b")
head(r_data$dat)
rm(r_data, envir = .GlobalEnv)
```

| | |
|------|-----------------------|
| city | <i>City distances</i> |
|------|-----------------------|

Description

City distances

Usage

```
data(city)
```

Format

A data frame with 45 rows and 3 variables

Details

Distance in miles between nine cities in the USA. The dataset is used to illustrate multi-dimensional scaling (MDS). Description provided in attr(city,"description")

| | |
|----------|--|
| ci_label | <i>Labels for confidence intervals</i> |
|----------|--|

Description

Labels for confidence intervals

Usage

```
ci_label(alt, cl)
```

Arguments

| | |
|-----|---|
| alt | Type of hypothesis ("two.sided", "less", "greater") |
| cl | Confidence level |

Value

A character vector with labels for a confidence interval

Examples

```
ci_label("less", .95)
ci_label("two.sided", .95)
ci_label("greater", .9)
```

| | |
|---------|------------------------------------|
| ci_perc | <i>Values at confidence levels</i> |
|---------|------------------------------------|

Description

Values at confidence levels

Usage

```
ci_perc(dat, alt, cl)
```

Arguments

| | |
|-----|---|
| dat | Data |
| alt | Type of hypothesis ("two.sided", "less", "greater") |
| cl | Confidence level |

Value

A character vector with labels for a confidence interval

Examples

```
ci_perc(0:100, "less", .95)
ci_perc(0:100, "greater", .95)
ci_perc(0:100, "two.sided", .80)
```

| | |
|----------------|--------------------------------|
| clean_loadings | <i>Sort and clean loadings</i> |
|----------------|--------------------------------|

Description

Sort and clean loadings

Usage

```
clean_loadings(floadings, cutoff = 0, fsort = FALSE, dec = 8)
```

Arguments

| | |
|-----------|--|
| floadings | Data frame with loadings |
| cutoff | Show only loadings with (absolute) values above cutoff (default = 0) |
| fsort | Sort factor loadings |
| dec | Number of decimals to show |

Details

See http://vnijs.github.io/radiant/marketing/full_factor.html for an example in Radiant

Examples

```
result <- full_factor("diamonds",c("price","carat","table","x","y"))
clean_loadings(result$floadings, TRUE, .5, 2)
```

| | |
|-------------|---|
| combinedata | <i>Combine datasets using dplyr's bind and join functions</i> |
|-------------|---|

Description

Combine datasets using dplyr's bind and join functions

Usage

```
combinedata(dataset, cmb_dataset, by = "", type = "inner_join", name = "")
```

Arguments

| | |
|-------------|---|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| cmb_dataset | Dataset name (string) to combine with 'dataset'. This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| by | Variables used to combine 'dataset' and 'cmb_dataset' |
| type | The main bind and join types from the dplyr package are provided. inner_join returns all rows from x with matching values in y, and all columns from x and y. If there are multiple matches between x and y, all match combinations are returned. left_join returns all rows from x, and all columns from x and y. If there are multiple matches between x and y, all match combinations are returned. right_join is equivalent to a left join for datasets y and x. full_join combines two datasets, keeping rows and columns that appear in either. semi_join returns all rows from x with matching values in y, keeping just columns from x. A semi join differs from an inner join because an inner join will return one row of x for each matching row of y, whereas a semi join will never duplicate rows of x. anti_join returns all rows from x without matching values in y, keeping only columns from x. bind_rows and bind_cols are also included, as are intersect , union , and setdiff . See http://vnijs.github.io/radiant/base/combine.html for further details |
| name | Name for the combined dataset |

Details

See <http://vnijs.github.io/radiant/base/combine.html> for an example in Radiant

Value

If list 'r_data' exists the combined dataset is added as 'name'. Else the combined dataset will be returned as 'name'

Examples

```
combinedata("titanic","titanic_pred",c("pclass","sex","age")) %>% head
titanic %>% combinedata("titanic_pred",c("pclass","sex","age")) %>% head
titanic %>% combinedata(titanic_pred,c("pclass","sex","age")) %>% head
avengers %>% combinedata(superheroes, type = "bind_cols")
combinedata("avengers", "superheroes", type = "bind_cols")
avengers %>% combinedata(superheroes, type = "bind_rows")
```

compare_means

*Compare means for two or more variables***Description**

Compare means for two or more variables

Usage

```
compare_means(dataset, var1, var2, samples = "independent",
  alternative = "two.sided", conf_lev = 0.95, comb = "",
  adjust = "none", test = "t", data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| var1 | A numeric variable or factor selected for comparison |
| var2 | One or more numeric variables for comparison. If var1 is a factor only one variable can be selected and the mean of this variable is compared across (factor) levels of var1 |
| samples | Are samples indepent ("independent") or not ("paired") |
| alternative | The alternative hypothesis ("two.sided", "greater" or "less") |
| conf_lev | Span of the confidence interval |
| comb | Combinations to evaluate |
| adjust | Adjustment for multiple comparisons ("none" or "bonf" for Bonferroni) |
| test | T-test ("t") or Wilcox ("wilcox") |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/quant/compare_means.html for an example in Radiant

Value

A list of all variables defined in the function as an object of class `compare_means`

See Also

[summary.compare_means](#) to summarize results

[plot.compare_means](#) to plot results

Examples

```
result <- compare_means("diamonds", "cut", "price")
result <- diamonds %>% compare_means("cut", "price")
```

compare_props

Compare proportions across groups

Description

Compare proportions across groups

Usage

```
compare_props(dataset, var1, var2, levs = "", alternative = "two.sided",
  conf_lev = 0.95, comb = "", adjust = "none", data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| var1 | A grouping variable to split the data for comparisons |
| var2 | The variable to calculate proportions for |
| levs | The factor level selected for the proportion comparison |
| alternative | The alternative hypothesis ("two.sided", "greater" or "less") |
| conf_lev | Span of the confidence interval |
| comb | Combinations to evaluate |
| adjust | Adjustment for multiple comparisons ("none" or "bonf" for Bonferroni) |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/quant/compare_props.html for an example in Radiant

Value

A list of all variables defined in the function as an object of class `compare_props`

See Also

[summary.compare_props](#) to summarize results

[plot.compare_props](#) to plot results

Examples

```
result <- compare_props("titanic", "pclass", "survived")
result <- titanic %>% compare_props("pclass", "survived")
```

| | |
|----------|--|
| computer | <i>Perceptions of computer (re)sellers</i> |
|----------|--|

Description

Perceptions of computer (re)sellers

Usage

```
data(computer)
```

Format

A data frame with 5 rows and 8 variables

Details

Perceptions of computer (re)sellers. The dataset is used to illustrate perceptual maps. Description provided in `attr(computer,"description")`

| | |
|----------|--------------------------|
| conjoint | <i>Conjoint analysis</i> |
|----------|--------------------------|

Description

Conjoint analysis

Usage

```
conjoint(dataset, dep_var, indep_var, reverse = FALSE, data_filter = "")
```

Arguments

| | |
|-------------|---|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| dep_var | The dependent variable (e.g., profile ratings) |
| indep_var | Independent variables in the regression |
| reverse | Reverse the values of the dependent variable ('dep_var') |
| data_filter | Expression entered in, e.g., <code>Data > View</code> to filter the dataset in Radiant. The expression should be a string (e.g., <code>"price > 10000"</code>) |

Details

See <http://vnijs.github.io/radiant/marketing/conjoint.html> for an example in Radiant

Value

A list with all variables defined in the function as an object of class `conjoint`

See Also

[summary.conjoint](#) to summarize results

[plot.conjoint](#) to plot results

Examples

```
result <- conjoint("mp3", dep_var = "Rating", indep_var = "Memory:Shape")
result <- mp3 %>% conjoint(dep_var = "Rating", indep_var = "Memory:Shape")
```

| | |
|-------------------|---|
| conjoint_profiles | <i>Create fractional factorial design for conjoint analysis</i> |
|-------------------|---|

Description

Create fractional factorial design for conjoint analysis

Usage

```
conjoint_profiles(dataset)
```

Arguments

| | |
|---------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
|---------|--|

Details

See http://vnijs.github.io/radiant/marketing/conjoint_profiles.html for an example in Radiant

Value

A list with all variables defined in the function as an object of class `conjoint_profiles`

See Also

[summary.conjoint_profiles](#) to summarize results

Examples

```
cp <- c("price = c('$10','$13','$16')", "sight = c('Staggered','Not Staggered')",
      "comfort = c('Average no cupholder','Average cupholder','Large cupholder')",
      "audio.visual = c('Small plain','Large plain','Large digital')",
      "food = c('No food','Hot dogs and popcorn','Gourmet food')")
result <- conjoint_profiles("cp")
result <- cp %>% conjoint_profiles
rm(cp, envir = .GlobalEnv)
```

| | |
|----------|-------------------------------------|
| copy_all | <i>Source all package functions</i> |
|----------|-------------------------------------|

Description

Source all package functions

Usage

```
copy_all(.from)
```

Arguments

| | |
|-------|---------------------------------------|
| .from | The package to pull the function from |
|-------|---------------------------------------|

Details

Equivalent of source with local=TRUE for all package functions. Adapted from functions by smbache, author of the import package. See <https://github.com/smbache/import/issues/4> for a discussion. This function will be deprecated when (if) it is included in <https://github.com/smbache/import>

Examples

```
copy_all(radiant)
```

| | |
|-----------|-------------------------------------|
| copy_from | <i>Source for package functions</i> |
|-----------|-------------------------------------|

Description

Source for package functions

Usage

```
copy_from(.from, ...)
```

Arguments

| | |
|-------|---------------------------------------|
| .from | The package to pull the function from |
| ... | Functions to pull |

Details

Equivalent of source with local=TRUE for package functions. Written by smbache, author of the import package. See <https://github.com/smbache/import/issues/4> for a discussion. This function will be deprecated when (if) it is included in <https://github.com/smbache/import>

Examples

```
copy_from(radiant, state_init)
```

| | |
|-------------|---|
| correlation | <i>Calculate correlations for two or more variables</i> |
|-------------|---|

Description

Calculate correlations for two or more variables

Usage

```
correlation(dataset, vars, type = "pearson", data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| vars | Variables to include in the analysis |
| type | Type of correlations to calculate. Options are "pearson", "spearman", and "kendall". "pearson" is the default |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See <http://vnijs.github.io/radiant/quant/correlation.html> for an example in Radiant

Value

A list with all variables defined in the function as an object of class `compare_means`

See Also

[summary.correlation_](#) to summarize results

[plot.correlation_](#) to plot results

Examples

```
result <- correlation("diamonds", c("price","carat"))
result <- correlation("diamonds", c("price","carat","clarity"))
result <- correlation("diamonds", "price:table")
result <- diamonds %>% correlation("price:table")
```

`cross_tabs`*Evaluate associations between categorical variables*

Description

Evaluate associations between categorical variables

Usage

```
cross_tabs(dataset, var1, var2, data_filter = "")
```

Arguments

| | |
|--------------------------|--|
| <code>dataset</code> | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| <code>var1</code> | A categorical variable |
| <code>var2</code> | Another categorical variable |
| <code>data_filter</code> | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/quant/cross_tabs.html for an example in Radiant

Value

A list of all variables used in `cross_tabs` as an object of class `cross_tabs`

See Also

[`summary.cross_tabs`](#) to summarize results

[`plot.cross_tabs`](#) to plot results

Examples

```
result <- cross_tabs("newspaper", "Income", "Newspaper")
result <- newspaper %>% cross_tabs("Income", "Newspaper")
```

| | |
|----|---------------------------------|
| cv | <i>Coefficient of variation</i> |
|----|---------------------------------|

Description

Coefficient of variation

Usage

```
cv(x, na.rm = TRUE)
```

Arguments

| | |
|-------|---|
| x | Input variable |
| na.rm | If TRUE missing values are removed before calculation |

Value

Coefficient of variation

Examples

```
cv(runif (100))
```

| | |
|--------------|-----------------------|
| decile_split | <i>Create deciles</i> |
|--------------|-----------------------|

Description

Create deciles

Usage

```
decile_split(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Factor variable

diamonds*Diamond prices*

Description

Diamond prices

Usage

```
data(diamonds)
```

Format

A data frame with 3000 rows and 10 variables

Details

A sample of 3,000 from the diamonds dataset bundled with ggplot2. Description provided in `attr(diamonds,"description")`

does_vary*Does a vector have non-zero variability?*

Description

Does a vector have non-zero variability?

Usage

```
does_vary(x)
```

Arguments

x Input variable

Value

Logical. TRUE if there is variability

Examples

```
summarise_each(diamonds, funs(does_vary)) %>% as.logical
```

| | |
|-------|-------------------------------|
| dtree | <i>Create a decision tree</i> |
|-------|-------------------------------|

Description

Create a decision tree

Usage

```
dtree(yl, opt = "max")
```

Arguments

| | |
|-----|---|
| yl | A yaml string or a list (e.g., from <code>yaml::yaml.load_file()</code>) |
| opt | Find the maximum ("max") or minimum ("min") value for each decision node |

Details

See <http://vnijs.github.io/radiant/base/dtree.html> for an example in Radiant

Value

A list with the initial tree and the calculated tree

See Also

[summary.dtree](#) to summarize results
[plot.dtree](#) to plot results

| | |
|--------------|---|
| dtree_parser | <i>Parse yaml input for dtree to provide (more) useful error messages</i> |
|--------------|---|

Description

Parse yaml input for dtree to provide (more) useful error messages

Usage

```
dtree_parser(yl)
```

Arguments

| | |
|----|---------------|
| yl | A yaml string |
|----|---------------|

Details

See <http://vnijs.github.io/radiant/base/dtree.html> for an example in Radiant

Value

An updated yaml string or a vector messages to return to the users

See Also

`dtree` to calculate tree
`summary.dtree` to summarize results
`plot.dtree` to plot results

| | |
|---------|---------------------|
| explore | <i>Explore data</i> |
|---------|---------------------|

Description

Explore data

Usage

```
explore(dataset, vars = "", byvar = "", fun = "mean_rm", tabfilt = "",
        tabsort = "", data_filter = "", shiny = FALSE)
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| vars | (Numerical) variables to summaries |
| byvar | Variable(s) to group data by before summarizing |
| fun | Functions to use for summarizing |
| tabfilt | Expression used to filter the table. This should be a string (e.g., "Total > 10000") |
| tabsort | Expression used to sort the table (e.g., "-Total") |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |
| shiny | Logical (TRUE, FALSE) to indicate if the function call originate inside a shiny app |

Details

See <http://vnijs.github.io/radiant/base/explore.html> for an example in Radiant

Value

A list of all variables defined in the function as an object of class `explore`

See Also

`summary.explore` to show summaries

Examples

```
result <- explore("diamonds", "price:x")
summary(result)
result <- explore("diamonds", c("price", "carat"), byvar = "cut", fun = c("n_missing", "skew"))
summary(result)
diamonds %>% explore("price", byvar = "cut", fun = c("length", "n_distinct"))
```

| | |
|------------|---|
| factorizer | <i>Convert character to factors as needed</i> |
|------------|---|

Description

Convert character to factors as needed

Usage

```
factorizer(dat, safx = 20)
```

Arguments

| | |
|-------|------------------------|
| dat | Data.frame |
| safox | Values to levels ratio |

Value

Data.frame with factors

| | |
|-----------|---|
| ff_design | <i>Function to generate a fractional factorial design</i> |
|-----------|---|

Description

Function to generate a fractional factorial design

Usage

```
ff_design(attr, trial = 0, rseed = 172110)
```

Arguments

| | |
|-------|---|
| attr | Attributes used to generate profiles |
| trial | Number of trials that have already been run |
| rseed | Random seed to use |

Details

See http://vnijs.github.io/radiant/marketing/conjoint_profiles.html for an example in Radiant

See Also

[conjoint_profiles](#) to calculate results
[summary.conjoint_profiles](#) to summarize results

| | |
|------------|---|
| filterdata | <i>Filter data with user-specified expression</i> |
|------------|---|

Description

Filter data with user-specified expression

Usage

```
filterdata(dat, filt = "")
```

Arguments

| | |
|------|--|
| dat | Data.frame to filter |
| filt | Filter expression to apply to the specified dataset (e.g., "price > 10000" if dataset is "diamonds") |

Value

Filtered data.frame

| | |
|------|--|
| flip | <i>Flip the DT table to put Function, Variable, or Group by on top</i> |
|------|--|

Description

Flip the DT table to put Function, Variable, or Group by on top

Usage

```
flip(expl, top = "fun")
```

Arguments

| | |
|------|---|
| expl | Return value from explore |
| top | The variable (type) to display at the top of the table ("fun" for Function, "var" for Variable, and "byvar" for Group by. "fun" is the default) |

Details

See <http://vnijs.github.io/radiant/base/explore.html> for an example in Radiant

See Also

[explore](#) to generate summaries
[make_expl](#) to create the DT table

Examples

```
result <- explore("diamonds", "price:x") %>% flip("var")

result <- explore("diamonds", "price", byvar = "cut", fun = c("length", "skew")) %>%
  flip("byvar")
```

full_factor

*Factor analysis (PCA)***Description**

Factor analysis (PCA)

Usage

```
full_factor(dataset, vars, method = "PCA", nr_fact = 2,
  rotation = "varimax", data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| vars | Variables to include in the analysis |
| method | Factor extraction method to use |
| nr_fact | Number of factors to extract |
| rotation | Apply varimax rotation or no rotation ("varimax" or "none") |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/marketing/full_factor.html for an example in Radiant

Value

A list with all variables defined in the function as an object of class `full_factor`

See Also

`summary.full_factor` to summarize results
`plot.full_factor` to plot results

Examples

```
result <- full_factor("diamonds",c("price","carat","table","x","y"))
result <- full_factor("diamonds",c("price","carat","table","x","y"), method = "maxlik")
result <- diamonds %>% full_factor(c("price","carat","table","x","y"), method = "maxlik")
```

`getclass`*Get variable class*

Description

Get variable class

Usage

```
getclass(dat)
```

Arguments

`dat` Dataset to evaluate

Details

Get variable class information for each column in a data.frame

Value

Vector with class information for each variable

Examples

```
getclass(mtcars)
```

`getdata`*Get data for analysis functions*

Description

Get data for analysis functions

Usage

```
getdata(dataset, vars = "", filt = "", rows = NULL, na.rm = TRUE)
```

Arguments

| | |
|----------------------|---|
| <code>dataset</code> | Name of the dataframe |
| <code>vars</code> | Variables to extract from the dataframe |
| <code>filt</code> | Filter to apply to the specified dataset. For example "price > 10000" if dataset is "diamonds" (default is "") |
| <code>rows</code> | Select rows in the specified dataset. For example "1:10" for the first 10 rows or "n()-10:n()" for the last 10 rows (default is NULL) |
| <code>na.rm</code> | Remove rows with missing values (default is TRUE) |

Value

Data.frame with specified columns and rows

Examples

```
r_data <- list()
r_data$dat <- mtcars
getdata("dat", "mpg:vs", filt = "mpg > 20", rows = 1:5)
rm(r_data, envir = .GlobalEnv)
```

| | |
|------------|----------------------------------|
| getsummary | <i>Create data.frame summary</i> |
|------------|----------------------------------|

Description

Create data.frame summary

Usage

```
getsummary(dat, dc = getclass(dat))
```

Arguments

| | |
|-----|-------------------------|
| dat | Data.frame |
| dc | Class for each variable |

Details

Used in Radiant's Data > Transform tab

| | |
|---------|--|
| glm_reg | <i>Generalized linear models (GLM)</i> |
|---------|--|

Description

Generalized linear models (GLM)

Usage

```
glm_reg(dataset, dep_var, indep_var, lev = "", link = "logit",
  int_var = "", check = "", data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| dep_var | The dependent variable in the logit (probit) model |
| indep_var | Independent variables in the model |
| lev | The level in the dependent variable defined as <code>_success_</code> |
| link | Link function for <code>_glm_</code> ('logit' or 'probit'). 'logit' is the default |
| int_var | Interaction term to include in the model (not implement) |
| check | Optional output or estimation parameters. "vif" to show the multicollinearity diagnostics. "confint" to show coefficient confidence interval estimates. "odds" to show odds ratios and confidence interval estimates. "standardize" to output standardized coefficient estimates. "stepwise" to apply step-wise selection of variables |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/quant/glm_reg.html for an example in Radiant

Value

A list with all variables defined in `glm_reg` as an object of class `glm_reg`

See Also

`summary.glm_reg` to summarize the results
`plot.glm_reg` to plot the results
`predict.glm_reg` to generate predictions
`plot.glm_predict` to plot prediction output

Examples

```
result <- glm_reg("titanic", "survived", c("pclass","sex"), lev = "Yes")
result <- glm_reg("titanic", "survived", c("pclass","sex"))
```

 hier_clus

Hierarchical cluster analysis

Description

Hierarchical cluster analysis

Usage

```
hier_clus(dataset, vars, distance = "sq.euclidian", method = "ward.D",
  data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| vars | Vector of variables to include in the analysis |
| distance | Distance |
| method | Method |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/marketing/hier_clus.html for an example in Radiant

Value

A list of all variables used in `hier_clus` as an object of class `hier_clus`

See Also

[summary.hier_clus](#) to summarize results

[plot.hier_clus](#) to plot results

Examples

```
result <- hier_clus("shopping", vars = c("v1:v6"))
```

inverse

Calculate inverse of a variable

Description

Calculate inverse of a variable

Usage

```
inverse(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

1/x

| | |
|----------|--|
| is_empty | <i>Is a character variable defined</i> |
|----------|--|

Description

Is a character variable defined

Usage

```
is_empty(x, empty = "")
```

Arguments

| | |
|-------|---|
| x | Character value to evaluate |
| empty | Indicate what 'empty' means. Default is empty string (i.e., "") |

Details

Is a variable NULL or an empty string

Value

TRUE if empty, else FALSE

Examples

```
is_empty("")  
is_empty(NULL)
```

| | |
|-----------|---------------------------|
| is_string | <i>Is input a string?</i> |
|-----------|---------------------------|

Description

Is input a string?

Usage

```
is_string(x)
```

Arguments

| | |
|---|-------|
| x | Input |
|---|-------|

Details

Is input a string

Value

TRUE if string, else FALSE

Examples

```
is_string("")
is_string("data")
is_string(c("data","data"))
is_string(NULL)
```

| | |
|--------|--------------------------------------|
| iterms | Create a vector of interaction terms |
|--------|--------------------------------------|

Description

Create a vector of interaction terms

Usage

```
iterms(vars, nway, sep = ":")
```

Arguments

| | |
|------|--|
| vars | Variables lables to use |
| nway | 2-way (2) or 3-way (3) interactions labels to create |
| sep | Separator between variable names (default is :) |

Value

Character vector of interaction term labels

Examples

```
paste0("var", 1:3) %>% iterms(2)
paste0("var", 1:3) %>% iterms(3)
paste0("var", 1:3) %>% iterms(2, sep = ".")
```

| | |
|-------------|--------------------------|
| kmeans_clus | K-means cluster analysis |
|-------------|--------------------------|

Description

K-means cluster analysis

Usage

```
kmeans_clus(dataset, vars, hc_init = TRUE, distance = "sq.euclidian",
  method = "ward.D", seed = 1234, nr_clus = 2, data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| vars | Vector of variables to include in the analysis |
| hc_init | Use centers from <code>hier_clus</code> as the starting point |
| distance | Distance for <code>hier_clus</code> |
| method | Method for <code>hier_clus</code> |
| seed | Random seed to use for <code>kmeans</code> if <code>hc_init</code> is FALSE |
| nr_clus | Number of clusters to extract |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/marketing/kmeans_clus.html for an example in Radiant

Value

A list of all variables used in `kmeans_clus` as an object of class `kmeans_clus`

See Also

[summary.kmeans_clus](#) to summarize results
[plot.kmeans_clus](#) to plot results
[save_membership](#) to add cluster membership to the selected dataset

Examples

```
result <- kmeans_clus("shopping", c("v1:v6"))
```

kurtosi

Exporting the kurtosi function from the psych package

Description

Exporting the `kurtosi` function from the `psych` package

| | |
|----------|--|
| launcher | <i>Create a launcher on the desktop for Windows (.bat), Mac (.command), or Linux (.sh)</i> |
|----------|--|

Description

Create a launcher on the desktop for Windows (.bat), Mac (.command), or Linux (.sh)

Usage

```
launcher(app = c("analytics", "marketing", "quant", "base"))
```

Arguments

| | |
|-----|---|
| app | App to run when the desktop icon is double-clicked ("analytics", "marketing", "quant", or "base"). Default is "analytics" |
|-----|---|

Details

On Windows/Mac/Linux a file named radiant.bat/radiant.command/radiant.sh will be put on the desktop. Double-click the file to launch the specified Radiant app

See Also

[win_launcher](#) to create a shortcut on Windows

[mac_launcher](#) to create a shortcut on Mac

[lin_launcher](#) to create a shortcut on Linux

| | |
|--------------|--|
| lin_launcher | <i>Create a launcher and updater for Linux (.sh)</i> |
|--------------|--|

Description

Create a launcher and updater for Linux (.sh)

Usage

```
lin_launcher(app = c("analytics", "marketing", "quant", "base"))
```

Arguments

| | |
|-----|---|
| app | App to run when the desktop icon is double-clicked ("analytics", "marketing", "quant", or "base"). Default is "analytics" |
|-----|---|

Details

On Linux a file named 'radiant.sh' and one named 'update_radiant.sh' will be put on the desktop. Double-click the file to launch the specified Radiant app or update Radiant to the latest version

Examples

```

if (interactive()) {
  if (Sys.info()["sysname"] == "Linux") {
    lin_launcher()
    fn <- paste0("/home/", Sys.getenv("USER"), "/Desktop/radiant.sh")
    if (!file.exists(fn))
      stop("Linux launcher not created")
    else
      unlink(fn)
  }
}

```

| | |
|---------|---|
| loadcsv | <i>Load a csv file with read.csv and read_csv</i> |
|---------|---|

Description

Load a csv file with read.csv and read_csv

Usage

```
loadcsv(fn, header = TRUE, sep = ",", dec = ".", saf = TRUE,
        safx = 20)
```

Arguments

| | |
|--------|---|
| fn | File name string |
| header | Header in file (TRUE, FALSE) |
| sep | Use , (default) or ; or \t |
| dec | Decimal symbol. Use . (default) or , |
| saf | Convert character variables to factors if (1) there are less than 100 distinct values (2) there are X (see safx) more values than levels |
| safx | Values to levels ratio |

Value

Data.frame with (some) variables converted to factors

| | |
|-------------|--|
| loadcsv_url | <i>Load a csv file with from a url</i> |
|-------------|--|

Description

Load a csv file with from a url

Usage

```
loadcsv_url(csv_url, header = TRUE, sep = ",", dec = ".", saf = TRUE,
            safx = 20)
```

Arguments

| | |
|---------|---|
| csv_url | URL for the csv file |
| header | Header in file (TRUE, FALSE) |
| sep | Use , (default) or ; or \t |
| dec | Decimal symbol. Use . (default) or , |
| saf | Convert character variables to factors if (1) there are less than 100 distinct values (2) there are X (see safx) more values than levels |
| safx | Values to levels ratio |

Value

Data.frame with (some) variables converted to factors

| | |
|---------|--|
| loadrda | <i>Load an rda file and add it to the radiant data list (r_data)</i> |
|---------|--|

Description

Load an rda file and add it to the radiant data list (r_data)

Usage

```
loadrda(fn, ext = "rda")
```

Arguments

| | |
|-----|---------------------------------------|
| fn | File name string |
| ext | File extension ("rda" is the default) |

Value

Data.frame in r_data

| | |
|-------------|------------------------------------|
| loadrda_url | <i>Load an rda file from a url</i> |
|-------------|------------------------------------|

Description

Load an rda file from a url

Usage

```
loadrda_url(rda_url)
```

Arguments

| | |
|---------|----------------------|
| rda_url | URL for the csv file |
|---------|----------------------|

Value

Data.frame

| | |
|--------------|---|
| mac_launcher | <i>Create a launcher and updater for Mac (.command)</i> |
|--------------|---|

Description

Create a launcher and updater for Mac (.command)

Usage

```
mac_launcher(app = c("analytics", "marketing", "quant", "base"))
```

Arguments

| | |
|-----|---|
| app | App to run when the desktop icon is double-clicked ("analytics", "marketing", "quant", or "base"). Default is "analytics" |
|-----|---|

Details

On Mac a file named 'radiant.command' and one named 'update_radiant.command' will be put on the desktop. Double-click the file to launch the specified Radiant app or update Radiant to the latest version

Examples

```
if (interactive()) {
  if (Sys.info()["sysname"] == "Darwin") {
    mac_launcher()
    fn <- paste0("/Users/", Sys.getenv("USER"), "/Desktop/radiant.command")
    if (!file.exists(fn))
      stop("Mac launcher not created")
    else
      unlink(fn)
  }
}
```

| | |
|---------|---------------------------------|
| make_dt | <i>Make a pivot tabel in DT</i> |
|---------|---------------------------------|

Description

Make a pivot tabel in DT

Usage

```
make_dt(pvt, format = "none", perc = FALSE, search = "",
        searchCols = NULL, order = NULL)
```

Arguments

| | |
|------------|---|
| pvt | Return value from pivotr |
| format | Show Color bar ("color_bar"), Heat map ("heat"), or None ("none") |
| perc | Display numbers as percentages (TRUE or FALSE) |
| search | Global search. Used to save and restore state |
| searchCols | Column search and filter. Used to save and restore state |
| order | Column sorting. Used to save and restore state |

Details

See <http://vnijs.github.io/radiant/base/pivotr.html> for an example in Radiant

See Also

[pivotr](#) to create the pivot-table using dplyr
[summary.pivotr](#) to print a plain text table

Examples

```
pivotr("diamonds", cvars = "cut") %>% make_dt
pivotr("diamonds", cvars = c("cut","clarity")) %>% make_dt(format = "color_bar")
ret <- pivotr("diamonds", cvars = c("cut","clarity"), normalize = "total") %>%
  make_dt(format = "color_bar", perc = TRUE)
```

| | |
|-----------|---|
| make_expl | <i>Make a tabel of summary statistics in DT</i> |
|-----------|---|

Description

Make a tabel of summary statistics in DT

Usage

```
make_expl(expl, top = "fun", dec = 3, search = "", searchCols = NULL,
          order = NULL)
```

Arguments

| | |
|------------|---|
| expl | Return value from explore |
| top | The variable (type) to display at the top of the table ("fun" for Function, "var" for Variable, and "byvar" for Group by) |
| dec | Number of decimals to show |
| search | Global search. Used to save and restore state |
| searchCols | Column search and filter. Used to save and restore state |
| order | Column sorting. Used to save and restore state |

Details

See <http://vnijs.github.io/radiant/base/explore.html> for an example in Radiant

See Also

[pivotr](#) to create the pivot-table using dplyr

[summary.pivotr](#) to print a plain text table

Examples

```
tab <- explore("diamonds", "price:x") %>% make_expl
tab <- explore("diamonds", "price", byvar = "cut", fun = c("length", "skew")) %>%
  make_expl(top = "byvar")
```

make_funs

Make a list of functions-as-formulas to pass to dplyr

Description

Make a list of functions-as-formulas to pass to dplyr

Usage

```
make_funs(x)
```

Arguments

| | |
|---|------------------------------|
| x | List of functions as strings |
|---|------------------------------|

Value

List of functions to pass to dplyr in formula form

Examples

```
make_funs(c("mean", "sum_rm"))
```

| | |
|------------|---|
| make_train | <i>Generate a variable used to selected a training sample</i> |
|------------|---|

Description

Generate a variable used to selected a training sample

Usage

```
make_train(n = 0.7, nr = 100)
```

Arguments

| | |
|----|---|
| n | Number (or fraction) of observations to label as training |
| nr | Number of rows in the dataset |

Value

0/1 variables for filtering

Examples

```
make_train(.5, 10)
```

| | |
|--------|------------------------------|
| max_rm | <i>Max with na.rm = TRUE</i> |
|--------|------------------------------|

Description

Max with na.rm = TRUE

Usage

```
max_rm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Maximum value

Examples

```
max_rm(runif (100))
```

| | |
|-----|--|
| mds | (Dis)similarity based brand maps (MDS) |
|-----|--|

Description

(Dis)similarity based brand maps (MDS)

Usage

```
mds(dataset, id1, id2, dis, method = "metric", nr_dim = 2,  
      data_filter = "")
```

Arguments

| | |
|-------------|---|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| id1 | A character variable or factor with unique entries |
| id2 | A character variable or factor with unique entries |
| dis | A numeric measure of brand dissimilarity |
| method | Apply metric or non-metric MDS |
| nr_dim | Number of dimensions |
| data_filter | Expression entered in, e.g., <code>Data > View</code> to filter the dataset in Radiant. The expression should be a string (e.g., <code>"price > 10000"</code>) |

Details

See <http://vnijs.github.io/radiant/marketing/mds.html> for an example in Radiant

Value

A list of all variables defined in the function as an object of class `mds`

See Also

[summary.mds](#) to summarize results

[plot.mds](#) to plot results

Examples

```
result <- mds("city", "from", "to", "distance")  
summary(result)  
result <- mds("diamonds", "clarity", "cut", "price")  
summary(result)
```

| | |
|---------|-------------------------------|
| mean_rm | <i>Mean with na.rm = TRUE</i> |
|---------|-------------------------------|

Description

Mean with na.rm = TRUE

Usage

```
mean_rm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Mean value

Examples

```
mean_rm(runif (100))
```

| | |
|-----------|---------------------------------|
| median_rm | <i>Median with na.rm = TRUE</i> |
|-----------|---------------------------------|

Description

Median with na.rm = TRUE

Usage

```
median_rm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Median value

Examples

```
median_rm(runif (100))
```

| | |
|--------------|---------------------|
| median_split | <i>Median split</i> |
|--------------|---------------------|

Description

Median split

Usage

```
median_split(x)
```

Arguments

x Input variable

Value

Factor variable deciles

| | |
|--------|------------------------------|
| min_rm | <i>Min with na.rm = TRUE</i> |
|--------|------------------------------|

Description

Min with na.rm = TRUE

Usage

```
min_rm(x)
```

Arguments

x Input variable

Value

Minimum value

Examples

```
min_rm(runif (100))
```

| | |
|---------|-------------------------------|
| mode_rm | <i>Mode with na.rm = TRUE</i> |
|---------|-------------------------------|

Description

Mode with na.rm = TRUE

Usage

```
mode_rm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Mode value

Examples

```
mode_rm(diamonds$cut)
```

| | |
|-----|--------------------------------------|
| mp3 | <i>Conjoint data for MP3 players</i> |
|-----|--------------------------------------|

Description

Conjoint data for MP3 players

Usage

```
data(mp3)
```

Format

A data frame with 18 rows and 6 variables

Details

Conjoint data for MP3 players. Description provided in attr(mp3,"description")

| | |
|-------------|--|
| mutate_each | <i>Add transformed variables to a data frame (NSE)</i> |
|-------------|--|

Description

Add tranformed variables to a data frame (NSE)

Usage

```
mutate_each(tbl, funs, ..., ext = "")
```

Arguments

| | |
|------|--|
| tbl | Data frame to add transformed variables to |
| funs | Function(s) to apply (e.g., funs(log)) |
| ... | Variables to transform |
| ext | Extension to add for each variable |

Details

Wrapper for `dplyr::mutate_each` that allows custom variable name extensions

Examples

```
mutate_each(mtcars, funs(log), mpg, cyl, ext = "_log")
```

| | |
|-----------|-----------------------------|
| newspaper | <i>Newspaper readership</i> |
|-----------|-----------------------------|

Description

Newspaper readership

Usage

```
data(newspaper)
```

Format

A data frame with 580 rows and 2 variables

Details

Newspaper readership data for 580 consumers. Description provided in `attr(newspaper,"description")`

| | |
|-----------|---|
| normalize | <i>Normalize a variable x by a variable y</i> |
|-----------|---|

Description

Normalize a variable x by a variable y

Usage

```
normalize(x, y)
```

Arguments

| | |
|---|----------------------|
| x | Input variable |
| y | Normalizing variable |

Value

x/y

| | |
|-----------|---------------------------------|
| n_missing | <i>Number of missing values</i> |
|-----------|---------------------------------|

Description

Number of missing values

Usage

```
n_missing(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

number of missing values

Examples

```
n_missing(c("a","b",NA))
```

| | |
|-----|----------------|
| p05 | 5th percentile |
|-----|----------------|

Description

5th percentile

Usage

```
p05(x, na.rm = TRUE)
```

Arguments

| | |
|-------|---|
| x | Input variable |
| na.rm | If TRUE missing values are removed before calculation |

Value

5th percentile

Examples

```
p05(rnorm(100))
```

| | |
|-----|-----------------|
| p25 | 25th percentile |
|-----|-----------------|

Description

25th percentile

Usage

```
p25(x, na.rm = TRUE)
```

Arguments

| | |
|-------|---|
| x | Input variable |
| na.rm | If TRUE missing values are removed before calculation |

Value

25th percentile

Examples

```
p25(rnorm(100))
```

| | |
|-----|-----------------|
| p75 | 75th percentile |
|-----|-----------------|

Description

75th percentile

Usage

```
p75(x, na.rm = TRUE)
```

Arguments

| | |
|-------|---|
| x | Input variable |
| na.rm | If TRUE missing values are removed before calculation |

Value

75th percentile

Examples

```
p75(rnorm(100))
```

| | |
|-----|-----------------|
| p95 | 95th percentile |
|-----|-----------------|

Description

95th percentile

Usage

```
p95(x, na.rm = TRUE)
```

Arguments

| | |
|-------|---|
| x | Input variable |
| na.rm | If TRUE missing values are removed before calculation |

Value

95th percentile

Examples

```
p95(rnorm(100))
```

pivotr

Create a pivot table using dplyr

Description

Create a pivot table using dplyr

Usage

```
pivotr(dataset, cvars = "", nvar = "None", fun = "mean",  
        normalize = "None", tabfilt = "", tabsort = "", data_filter = "",  
        shiny = FALSE)
```

Arguments

| | |
|-------------|--|
| dataset | Name of the dataframe to change |
| cvars | Categorical variables |
| nvar | Numerical variable |
| fun | Function to apply to numerical variable |
| normalize | Normalize the table by "row" total,"column" totals, or overall "total" |
| tabfilt | Expression used to filter the table. This should be a string (e.g., "Total > 10000") |
| tabsort | Expression used to sort the table (e.g., "-Total") |
| data_filter | Expression used to filter the dataset. This should be a string (e.g., "price > 10000") |
| shiny | Logical (TRUE, FALSE) to indicate if the function call originate inside a shiny app |

Details

Create a pivot-table. See <http://vnijs.github.io/radiant/base/pivotr.html> for an example in Radiant

Examples

```
result <- pivotr("diamonds", cvars = "cut")$tab  
result <- pivotr("diamonds", cvars = c("cut","clarity","color"))$tab  
result <- pivotr("diamonds", cvars = "cut:clarity", nvar = "price")$tab
```

| | |
|--------------------|---|
| plot.compare_means | <i>Plot method for the compare_means function</i> |
|--------------------|---|

Description

Plot method for the compare_means function

Usage

```
## S3 method for class 'compare_means'  
plot(x, plots = "scatter", shiny = FALSE, ...)
```

Arguments

| | |
|-------|---|
| x | Return value from compare_means |
| plots | One or more plots ("bar", "density", "box", or "scatter") |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/compare_means.html for an example in Radiant

See Also

[compare_means](#) to calculate results
[summary.compare_means](#) to summarize results

Examples

```
result <- compare_means("diamonds", "cut", "price")  
plot(result, plots = c("bar", "density"))
```

| | |
|--------------------|---|
| plot.compare_props | <i>Plot method for the compare_props function</i> |
|--------------------|---|

Description

Plot method for the compare_props function

Usage

```
## S3 method for class 'compare_props'  
plot(x, plots = "bar", shiny = FALSE, ...)
```


Arguments

| | |
|-------|---|
| x | Return value from compare_props |
| plots | One or more plots of proportions ("bar" or "dodge") |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/compare_props.html for an example in Radiant

See Also

[compare_props](#) to calculate results
[summary.compare_props](#) to summarize results

Examples

```
result <- compare_props("titanic", "pclass", "survived")
plot(result, plots = c("bar", "dodge"))
```

plot.conjoint

Plot method for the conjoint function

Description

Plot method for the conjoint function

Usage

```
## S3 method for class 'conjoint'
plot(x, plots = "pw", scale_plot = FALSE,
     shiny = FALSE, ...)
```

Arguments

| | |
|------------|---|
| x | Return value from conjoint |
| plots | Show either the part-worth ("pw") or importance-weights ("iw") plot |
| scale_plot | Scale the axes of the part-worth plots to the same range |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/marketing/conjoint.html> for an example in Radiant

See Also

[conjoint](#) to generate results
[summary.conjoint](#) to summarize results

Examples

```
result <- conjoint(dataset = "mp3", dep_var = "Rating", indep_var = "Memory:Shape")
plot(result, scale_plot = TRUE)
plot(result, plots = "iw")
```

| | |
|-------------------|---|
| plot.correlation_ | <i>Plot method for the correlation function</i> |
|-------------------|---|

Description

Plot method for the correlation function

Usage

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

Arguments

| | |
|-----|--|
| x | Return value from correlation |
| ... | further arguments passed to or from other methods. |

Details

See <http://vnijs.github.io/radiant/quant/correlation.html> for an example in Radiant

See Also

[correlation](#) to calculate results
[summary.correlation_](#) to summarize results

Examples

```
result <- correlation("diamonds",c("price","carat","clarity"))
plot(result)
diamonds %>% correlation("price:clarity") %>% plot
```

| | |
|-----------------|--|
| plot.cross_tabs | <i>Plot method for the cross_tabs function</i> |
|-----------------|--|

Description

Plot method for the cross_tabs function

Usage

```
## S3 method for class 'cross_tabs'
plot(x, check = "", shiny = FALSE, ...)
```

Arguments

| | |
|-------|---|
| x | Return value from cross_tabs |
| check | Show plots for variables var1 and var2. "observed" for the observed frequencies table, "expected" for the expected frequencies table (i.e., frequencies that would be expected if the null hypothesis holds), "chi_sq" for the contribution to the overall chi-squared statistic for each cell (i.e., $(o - e)^2 / e$), "dev_std" for the standardized differences between the observed and expected frequencies (i.e., $(o - e) / \sqrt{e}$), and "row_perc", "col_perc", and "perc" for row, column, and table percentages respectively |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/cross_tabs.html for an example in Radiant

See Also

[cross_tabs](#) to calculate results
[summary.cross_tabs](#) to summarize results

Examples

```
result <- cross_tabs("newspaper", "Income", "Newspaper")
plot(result, check = c("observed", "expected", "chi_sq"))
newspaper %>% cross_tabs("Income", "Newspaper") %>% plot(c("observed", "expected"))
```

plot.dtree

*Plot method for the dtree function***Description**

Plot method for the dtree function

Usage

```
## S3 method for class 'dtree'
plot(x, symbol = "$", dec = 3, final = FALSE,
     shiny = FALSE, ...)
```

Arguments

| | |
|--------|---|
| x | Return value from dtree |
| symbol | Monetary symbol to use (\$ is the default) |
| dec | Decimal places to round results to |
| final | If TRUE plot the decision tree solution, else the initial decision tree |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/dtree.html> for an example in Radiant

See Also

[dtree](#) to generate the result

[summary.dtree](#) to summarize results

| | |
|------------------|---|
| plot.full_factor | <i>Plot method for the full_factor function</i> |
|------------------|---|

Description

Plot method for the full_factor function

Usage

```
## S3 method for class 'full_factor'
plot(x, shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from full_factor |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/marketing/full_factor.html for an example in Radiant

See Also

[full_factor](#) to calculate results

[plot.full_factor](#) to plot results

Examples

```
result <- full_factor("diamonds",c("price","carat","table"))
plot(result)
result <- full_factor("computer","high_end:business")
summary(result)
```

| | |
|------------------|---|
| plot.glm_predict | <i>Plot method for the predict.glm_reg function</i> |
|------------------|---|

Description

Plot method for the predict.glm_reg function

Usage

```
## S3 method for class 'glm_predict'
plot(x, xvar = "", facet_row = ".", facet_col = ".",
     color = "none", conf_lev = 0.95, ...)
```

Arguments

| | |
|-----------|---|
| x | Return value from <code>predict.glm_reg</code> . |
| xvar | Variable to display along the X-axis of the plot |
| facet_row | Create vertically arranged subplots for each level of the selected factor variable |
| facet_col | Create horizontally arranged subplots for each level of the selected factor variable |
| color | Adds color to a scatter plot to generate a heat map. For a line plot one line is created for each group and each is assigned a different colour |
| conf_lev | Confidence level to use for prediction intervals (.95 is the default). Note that the error bars for predictions are approximations at this point. |
| ... | further arguments passed to or from other methods |

Details

See http://vnijis.github.io/radiant/quant/glm_reg.html for an example in Radiant

See Also

`glm_reg` to generate the result
`summary.glm_reg` to summarize results
`plot.glm_reg` to plot results
`predict.glm_reg` to generate predictions

Examples

```
result <- glm_reg("titanic", "survived", c("pclass","sex","age"), lev = "Yes")
pred <- predict(result, pred_cmd = "pclass = levels(pclass)")
plot(pred, xvar = "pclass")
pred <- predict(result, pred_cmd = "age = 0:100")
plot(pred, xvar = "age")
pred <- predict(result, pred_cmd = "pclass = levels(pclass), sex = levels(sex)")
plot(pred, xvar = "pclass", color = "sex")
pred <- predict(result, pred_cmd = "pclass = levels(pclass), age = seq(0,100,20)")
plot(pred, xvar = "pclass", color = "age")
plot(pred, xvar = "age", color = "pclass")
pred <- predict(result, pred_cmd="pclass=levels(pclass), sex=levels(sex), age=seq(0,100,20)")
```

```
plot(pred, xvar = "age", color = "sex", facet_col = "pclass")
plot(pred, xvar = "age", color = "pclass", facet_col = "sex")
pred <- predict(result, pred_cmd="pclass=levels(pclass), sex=levels(sex), age=seq(0,100,5)")
plot(pred, xvar = "age", color = "sex", facet_col = "pclass")
plot(pred, xvar = "age", color = "pclass", facet_col = "sex")
```

plot.glm_reg

Plot method for the glm_reg function

Description

Plot method for the glm_reg function

Usage

```
## S3 method for class 'glm_reg'
plot(x, plots = "", conf_lev = 0.95, intercept = FALSE,
     shiny = FALSE, ...)
```

Arguments

| | |
|-----------|---|
| x | Return value from glm_reg |
| plots | Plots to produce for the specified GLM model. Use "" to avoid showing any plots (default). "hist" shows histograms of all variables in the model. "scatter" shows scatter plots (or box plots for factors) for the dependent variable with each independent variable. "dashboard" is a series of four plots used to visually evaluate model. "coef" provides a coefficient plot |
| conf_lev | Confidence level to use for coefficient and odds confidence intervals (.95 is the default) |
| intercept | Include the intercept in the coefficient plot (TRUE or FALSE). FALSE is the default |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/glm_reg.html for an example in Radiant

See Also

[glm_reg](#) to generate results
[plot.glm_reg](#) to plot results
[predict.glm_reg](#) to generate predictions
[plot.glm_predict](#) to plot prediction output

Examples

```
result <- glm_reg("titanic", "survived", c("pclass","sex"), lev = "Yes")
plot(result, plots = "coef")
```

| | |
|----------------|---|
| plot.hier_clus | <i>Plot method for the hier_clus function</i> |
|----------------|---|

Description

Plot method for the hier_clus function

Usage

```
## S3 method for class 'hier_clus'
plot(x, plots = c("scree", "diff"), cutoff = 0.02,
     shiny = FALSE, ...)
```

Arguments

| | |
|--------|---|
| x | Return value from hier_clus |
| plots | Plots to return. "diff" shows the percentage change in within-cluster heterogeneity as respondents are group into different number of clusters, "dendro" shows the dendrogram, "scree" shows a scree plot of within-cluster heterogeneity |
| cutoff | For large datasets plots can take time to render and become hard to interpret. By selection a cutoff point (e.g., 0.05 percent) the initial steps in hierachical cluster analysis are removed from the plot |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/marketing/hier_clus.html for an example in Radiant

See Also

[summary.hier_clus](#) to summarize results

[plot.hier_clus](#) to plot results

Examples

```
result <- hier_clus("shopping", vars = c("v1:v6"))
plot(result, plots = c("diff", "scree"), cutoff = .05)
plot(result, plots = "dendro", cutoff = 0)
shopping %>% hier_clus(vars = c("v1:v6")) %>% plot
```

| | |
|------------------|------------------------------------|
| plot.kmeans_clus | <i>Plot method for kmeans_clus</i> |
|------------------|------------------------------------|

Description

Plot method for kmeans_clus

Usage

```
## S3 method for class 'kmeans_clus'  
plot(x, shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from kmeans_clus |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/marketing/kmeans_clus.html for an example in Radiant

See Also

[kmeans_clus](#) to generate results
[summary.kmeans_clus](#) to summarize results
[save_membership](#) to add cluster membership to the selected dataset

Examples

```
result <- kmeans_clus("shopping", vars = c("v1:v6"))  
plot(result)
```

| | |
|----------|---|
| plot.mds | <i>Plot method for the mds function</i> |
|----------|---|

Description

Plot method for the mds function

Usage

```
## S3 method for class 'mds'  
plot(x, rev_dim = "", fontsz = 1.3, ...)
```


Arguments

| | |
|---------|---|
| x | Return value from mds |
| rev_dim | Flip the axes in plots |
| fontsz | Font size to use in plots |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/marketing/mds.html> for an example in Radiant

See Also

[mds](#) to calculate results
[summary.mds](#) to plot results

Examples

```
result <- mds("city", "from", "to", "distance")
plot(result)
plot(result, rev_dim = 1:2)
plot(result, rev_dim = 1:2, fontsz = 2)
```

plot.pivotr

Plot method for the pivotr function

Description

Plot method for the pivotr function

Usage

```
## S3 method for class 'pivotr'
plot(x, type = "dodge", perc = FALSE, flip = FALSE,
      shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from pivotr |
| type | Plot type to use ("fill" or "dodge" (default)) |
| perc | Use percentage on the y-axis |
| flip | Flip the axes in a plot (FALSE or TRUE) |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/base/pivotr> for an example in Radiant

See Also

[pivotr](#) to generate summaries

[summary.pivotr](#) to show summaries

Examples

```

pivotr("diamonds", cvars = "cut") %>% plot
pivotr("diamonds", cvars = c("cut", "clarity")) %>% plot
pivotr("diamonds", cvars = c("cut", "clarity", "color")) %>% plot

```

plot.pmap

Plot method for the pmap function

Description

Plot method for the pmap function

Usage

```

## S3 method for class 'pmap'
plot(x, plots = "", scaling = 2.1, fontsz = 1.3, ...)

```

Arguments

| | |
|---------|--|
| x | Return value from pmap |
| plots | Components to include in the plot ("brand", "attr"). If data on preferences is available use "pref" to add preference arrows to the plot |
| scaling | Arrow scaling in the brand map |
| fontsz | Font size to use in plots |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/marketing/pmap.html> for an example in Radiant

See Also

[pmap](#) to calculate results

[summary.pmap](#) to plot results

Examples

```

result <- pmap("computer", "brand", "high_end:business")
plot(result, plots = "brand")
plot(result, plots = c("brand", "attr"))
plot(result, plots = c("brand", "attr"))
plot(result, scaling = 1, plots = c("brand", "attr"))
result <- pmap("computer", "brand", "high_end:dated",
               pref = c("innovative", "business"))
plot(result, plots = c("brand", "attr", "pref"))

```

| | |
|-----------------|--|
| plot.pre_factor | <i>Plot method for the pre_factor function</i> |
|-----------------|--|

Description

Plot method for the pre_factor function

Usage

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

Arguments

| | |
|-----|---|
| x | Return value from pre_factor |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/marketing/pre_factor.html for an example in Radiant

See Also

[pre_factor](#) to calculate results
[summary.pre_factor](#) to summarize results

Examples

```
result <- pre_factor("diamonds",c("price","carat","table"))
plot(result)
```

| | |
|-----------------|---|
| plot.prob_binom | <i>Plot method for the probability calculator function (binomial)</i> |
|-----------------|---|

Description

Plot method for the probability calculator function (binomial)

Usage

```
## S3 method for class 'prob_binom'
plot(x, type = "values", shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from prob_binom |
| type | Probabilities or values |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-----------------|--|
| plot.prob_chisq | <i>Plot method for the probability calculator (Chi-squared distribution)</i> |
|-----------------|--|

Description

Plot method for the probability calculator (Chi-squared distribution)

Usage

```
## S3 method for class 'prob_chisq'
plot(x, type = "values", shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from prob_chisq |
| type | Probabilities or values |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|----------------|---|
| plot.prob_disc | <i>Plot method for the probability calculator function (discrete)</i> |
|----------------|---|

Description

Plot method for the probability calculator function (discrete)

Usage

```
## S3 method for class 'prob_disc'
plot(x, type = "values", shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from prob_disc |
| type | Probabilities or values |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-----------------|--|
| plot.prob_fdist | <i>Plot method for the probability calculator (F-distribution)</i> |
|-----------------|--|

Description

Plot method for the probability calculator (F-distribution)

Usage

```
## S3 method for class 'prob_fdist'  
plot(x, type = "values", shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from prob_fdist |
| type | Probabilities or values |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|----------------|--|
| plot.prob_norm | <i>Plot method for the probability calculator (normal)</i> |
|----------------|--|

Description

Plot method for the probability calculator (normal)

Usage

```
## S3 method for class 'prob_norm'  
plot(x, type = "values", shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from prob_norm |
| type | Probabilities or values |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-----------------|--|
| plot.prob_tdist | <i>Plot method for the probability calculator (t-distribution)</i> |
|-----------------|--|

Description

Plot method for the probability calculator (t-distribution)

Usage

```
## S3 method for class 'prob_tdist'
plot(x, type = "values", shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from prob_tdist |
| type | Probabilities or values |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|----------------|---|
| plot.prob_unif | <i>Plot method for the probability calculator (uniform)</i> |
|----------------|---|

Description

Plot method for the probability calculator (uniform)

Usage

```
## S3 method for class 'prob_unif'
plot(x, type = "values", shiny = FALSE, ...)
```

Arguments

| | |
|-------|--|
| x | Return value from prob_unif |
| type | Probabilities or values |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-----------------|--|
| plot.regression | <i>Plot method for the regression function</i> |
|-----------------|--|

Description

Plot method for the regression function

Usage

```
## S3 method for class 'regression'
plot(x, plots = "", lines = "", conf_lev = 0.95,
     intercept = FALSE, shiny = FALSE, ...)
```

Arguments

| | |
|-----------|--|
| x | Return value from regression |
| plots | Regression plots to produce for the specified regression model. Enter "" to avoid showing any plots (default). "hist" to show histograms of all variables in the model. "correlations" for a visual representation of the correlation matrix selected variables. "scatter" to show scatter plots (or box plots for factors) for the dependent variables with each explanatory variable. "dashboard" for a series of six plots that can be used to evaluate model fit visually. "resid_pred" to plot the explanatory variables against the model residuals. "coef" for a coefficient plot with adjustable confidence intervals. "leverage" to show leverage plots for each explanatory variable |
| lines | Optional lines to include in the select plot. "line" to include a line through a scatter plot. "loess" to include a polynomial regression fit line. To include both use c("line","loess") |
| conf_lev | Confidence level used to estimate confidence intervals (.95 is the default) |
| intercept | Include the intercept in the coefficient plot (TRUE, FALSE). FALSE is the default |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/regression.html> for an example in Radiant

See Also

[regression](#) to generate the results
[summary.regression](#) to summarize results
[predict.regression](#) to generate predictions

Examples

```

result <- regression("diamonds", "price", c("carat","clarity"))
plot(result, plots = "dashboard")
plot(result, plots = "dashboard", lines = c("line","loess"))
plot(result, plots = "coef", intercept = TRUE)
plot(result, plots = "coef", conf_lev = .99, intercept = TRUE)
plot(result, plots = "hist")
plot(result, plots = "scatter", lines = c("line","loess"))
plot(result, plots = "correlations")
plot(result, plots = "leverage")
plot(result, plots = "resid_pred", lines = "line")

```

| | |
|------------------|--|
| plot.reg_predict | <i>Plot method for the predict.regression function</i> |
|------------------|--|

Description

Plot method for the predict.regression function

Usage

```

## S3 method for class 'reg_predict'
plot(x, xvar = "", facet_row = ".", facet_col = ".",
     color = "none", conf_lev = 0.95, ...)

```

Arguments

| | |
|-----------|---|
| x | Return value from predict.regression . |
| xvar | Variable to display along the X-axis of the plot |
| facet_row | Create vertically arranged subplots for each level of the selected factor variable |
| facet_col | Create horizontally arranged subplots for each level of the selected factor variable |
| color | Adds color to a scatter plot to generate a heat map. For a line plot one line is created for each group and each is assigned a different colour |
| conf_lev | Confidence level to use for prediction intervals (.95 is the default). Note that the error bars for predictions are approximations at this point. |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/regression.html> for an example in Radiant

See Also

[regression](#) to generate the result
[summary.regression](#) to summarize results
[plot.regression](#) to plot results
[predict.regression](#) to generate predictions

Examples

```

result <- regression("diamonds", "price", c("carat","clarity"))
pred <- predict(result, pred_cmd = "carat = 1:10")
plot(pred, xvar = "carat")
result <- regression("diamonds", "price", c("carat","clarity"), int_var = "carat:clarity")
dpred <- getdata("diamonds") %>% slice(1:100)
pred <- predict(result, pred_data = "dpred")
plot(pred, xvar = "carat", color = "clarity")
rm(dpred, envir = .GlobalEnv)

```

| | |
|---------------|---------------------------------|
| plot.repeater | <i>Plot repeated simulation</i> |
|---------------|---------------------------------|

Description

Plot repeated simulation

Usage

```

## S3 method for class 'repeater'
plot(x, sum_vars = "", byvar = "sim", fun = c("sum_rm",
      "mean_rm", "sd_rm"), shiny = FALSE, ...)

```

Arguments

| | |
|----------|--|
| x | Return value from repeater |
| sum_vars | (Numerical) variables to summaries |
| byvar | Variable(s) to group data by before summarizing |
| fun | Functions to use for summarizing |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

| | |
|----------------|---|
| plot.simulater | <i>Plot method for the simulater function</i> |
|----------------|---|

Description

Plot method for the simulater function

Usage

```

## S3 method for class 'simulater'
plot(x, shiny = FALSE, ...)

```

Arguments

| | |
|-------|--|
| x | Return value from simulater |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/simulater> for an example in Radiant

See Also

[single_mean](#) to generate the result
[summary.single_mean](#) to summarize results

Examples

```
result <- simulater(const = "cost 3", norm = "demand 2000 1000",
                    discrete = "price 5 .3 8 .7",
                    form = "profit = demand * (price - cost)")
plot(result)
```

| | |
|------------------|---|
| plot.single_mean | <i>Plot method for the single_mean function</i> |
|------------------|---|

Description

Plot method for the single_mean function

Usage

```
## S3 method for class 'single_mean'
plot(x, plots = "hist", shiny = FALSE, ...)
```

Arguments

| | |
|-------|---|
| x | Return value from single_mean |
| plots | Plots to generate. "hist" shows a histogram of the data along with vertical lines that indicate the sample mean and the confidence interval. "simulate" shows the location of the sample mean and the comparison value (comp_value). Simulation is used to demonstrate the sampling variability in the data under the null-hypothesis |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/single_mean.html for an example in Radiant

See Also

[single_mean](#) to generate the result
[summary.single_mean](#) to summarize results

Examples

```
result <- single_mean("diamonds", "price", comp_value = 3500)
plot(result, plots = c("hist", "simulate"))
```

| | |
|------------------|---|
| plot.single_prop | <i>Plot method for the single_prop function</i> |
|------------------|---|

Description

Plot method for the single_prop function

Usage

```
## S3 method for class 'single_prop'
plot(x, plots = "hist", shiny = FALSE, ...)
```

Arguments

| | |
|-------|---|
| x | Return value from single_prop |
| plots | Plots to generate. "hist" shows a histogram of the data along with vertical lines that indicate the sample proportion and the confidence interval. "simulate" shows the location of the sample proportion and the comparison value (comp_value). Simulation is used to demonstrate the sampling variability in the data under the null-hypothesis |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/single_prop.html for an example in Radiant

See Also

[single_prop](#) to generate the result
[summary.single_prop](#) to summarize the results

Examples

```
result <- single_prop("diamonds","clarity", lev = "IF", comp_value = 0.05)
plot(result, plots = c("hist", "simulate"))
result <- single_prop("titanic","pclass", lev = "1st")
plot(result, plots = c("hist","simulate"))
```

| | |
|------|-----------------------------------|
| pmap | <i>Attribute based brand maps</i> |
|------|-----------------------------------|

Description

Attribute based brand maps

Usage

```
pmap(dataset, brand, attr, pref = "", nr_dim = 2, data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| brand | A character variable with brand names |
| attr | Names of numeric variables |
| pref | Names of numeric brand preference measures |
| nr_dim | Number of dimensions |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See <http://vnijs.github.io/radiant/marketing/pmap.html> for an example in Radiant

Value

A list of all variables defined in the function as an object of class `pmap`

See Also

[summary.pmap](#) to summarize results

[plot.pmap](#) to plot results

Examples

```
result <- pmap("computer", "brand", "high_end:business")
```

| | |
|-----------------|--|
| predict.glm_reg | <i>Predict method for the glm_reg function</i> |
|-----------------|--|

Description

Predict method for the `glm_reg` function

Usage

```
## S3 method for class 'glm_reg'
predict(object, pred_vars = "", pred_data = "",
        pred_cmd = "", prn = TRUE, ...)
```

Arguments

| | |
|-----------|--|
| object | Return value from glm_reg |
| pred_vars | Variables selected to generate predictions |
| pred_data | Provide the name of a dataframe to generate predictions (e.g., "titanic"). The dataset must contain all columns used in the estimation |
| pred_cmd | Generate predictions using a command. For example, 'pclass = levels(pclass)' would produce predictions for the different levels of factor 'pclass'. To add another variable use a ',' (e.g., 'pclass = levels(pclass), age = seq(0,100,20)') |
| prn | Print prediction results (default is TRUE) |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/glm_reg.html for an example in Radiant

See Also

`glm_reg` to generate the result
`summary.glm_reg` to summarize results
`plot.glm_reg` to plot results
`plot.glm_predict` to plot prediction output

Examples

```
result <- glm_reg("titanic", "survived", c("pclass","sex"), lev = "Yes")
predict(result, pred_cmd = "pclass = levels(pclass)")
glm_reg("titanic", "survived", c("pclass","sex"), lev = "Yes") %>%
  predict(pred_cmd = "sex = c('male','female')")
glm_reg("titanic", "survived", c("pclass","sex"), lev = "Yes") %>%
  predict(pred_data = "titanic")
```

| | |
|--------------------|---|
| predict.regression | <i>Predict method for the regression function</i> |
|--------------------|---|

Description

Predict method for the regression function

Usage

```
## S3 method for class 'regression'
predict(object, pred_vars = "", pred_data = "",
  pred_cmd = "", conf_lev = 0.95, prn = TRUE, ...)
```

Arguments

| | |
|-----------|---|
| object | Return value from <code>regression</code> |
| pred_vars | Variables to use for prediction |
| pred_data | Name of the dataset to use for prediction |
| pred_cmd | Command used to generate data for prediction |
| conf_lev | Confidence level used to estimate confidence intervals (.95 is the default) |
| prn | Print prediction results (default is TRUE) |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/regression.html> for an example in Radiant

See Also

[regression](#) to generate the result
[summary.regression](#) to summarize results
[plot.regression](#) to plot results

Examples

```
result <- regression("diamonds", "price", c("carat","clarity"))
predict(result, pred_cmd = "carat = 1:10")
predict(result, pred_cmd = "clarity = levels(clarity)")
result <- regression("diamonds", "price", c("carat","clarity"), int_var = c("carat:clarity"))
dpred <- getdata("diamonds") %>% slice(1:10)
predict(result, pred_data = "dpred")
rm(dpred, envir = .GlobalEnv)
```

pre_factor

*Evaluate if data are appropriate for PCA / Factor analysis***Description**

Evaluate if data are appropriate for PCA / Factor analysis

Usage

```
pre_factor(dataset, vars, data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| vars | Variables to include in the analysis |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/marketing/pre_factor.html for an example in Radiant

Value

A list with all variables defined in the function as an object of class `pre_factor`

See Also

[summary.pre_factor](#) to summarize results
[plot.pre_factor](#) to plot results

Examples

```
result <- pre_factor("diamonds",c("price","carat","table"))
```

| | |
|--------------|--|
| print.gtable | <i>Print/draw method for grobs produced by gridExtra</i> |
|--------------|--|

Description

Print/draw method for grobs produced by gridExtra

Usage

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

Arguments

| | |
|-----|---|
| x | a gtable object |
| ... | further arguments passed to or from other methods |

Details

Print method for ggplot grobs created using arrangeGrob. Code is based on <https://github.com/baptiste/gridextra/blob/master/inst/testing/shiny.R>

Value

A plot

| | |
|------------|--|
| prob_binom | <i>Probability calculator for the binomial distribution (binomial)</i> |
|------------|--|

Description

Probability calculator for the binomial distribution (binomial)

Usage

```
prob_binom(n, p, lb = NA, ub = NA, plb = NA, pub = NA, dec = 3)
```

Arguments

| | |
|-----|--|
| n | Number of trials |
| p | Probability |
| lb | Lower bound on the number of successes |
| ub | Upper bound on the number of successes |
| plb | Lower probability bound |
| pub | Upper probability bound |
| dec | Number of decimals to show |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|------------|--|
| prob_chisq | <i>Probability calculator for the chi-squared distribution</i> |
|------------|--|

Description

Probability calculator for the chi-squared distribution

Usage

```
prob_chisq(df, lb = NA, ub = NA, plb = NA, pub = NA, dec = 3)
```

Arguments

| | |
|-----|-------------------------------|
| df | Degrees of freedom |
| lb | Lower bound (default is -Inf) |
| ub | Upper bound (default is Inf) |
| plb | Lower probability bound |
| pub | Upper probability bound |
| dec | Number of decimals to show |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-----------|--|
| prob_disc | <i>Probability calculator for the discrete distribution (discrete)</i> |
|-----------|--|

Description

Probability calculator for the discrete distribution (discrete)

Usage

```
prob_disc(v, p, lb = NA, ub = NA, plb = NA, pub = NA, dec = 3)
```

Arguments

| | |
|-----|--|
| v | Values |
| p | Probabilities |
| lb | Lower bound on the number of successes |
| ub | Upper bound on the number of successes |
| plb | Lower probability bound |
| pub | Upper probability bound |
| dec | Number of decimals to show |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|------------|--|
| prob_fdist | <i>Probability calculator for the F-distribution</i> |
|------------|--|

Description

Probability calculator for the F-distribution

Usage

```
prob_fdist(df1, df2, lb = NA, ub = NA, plb = NA, pub = NA, dec = 3)
```

Arguments

| | |
|-----|-------------------------------|
| df1 | Degrees of freedom |
| df2 | Degrees of freedom |
| lb | Lower bound (default is -Inf) |
| ub | Upper bound (default is Inf) |
| plb | Lower probability bound |
| pub | Upper probability bound |
| dec | Number of decimals to show |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-----------|---|
| prob_norm | <i>Probability calculator for the normal distribution</i> |
|-----------|---|

Description

Probability calculator for the normal distribution

Usage

```
prob_norm(mean, stdev, lb = NA, ub = NA, plb = NA, pub = NA, dec = 3)
```

Arguments

| | |
|-------|-------------------------------|
| mean | Mean |
| stdev | Standard deviation |
| lb | Lower bound (default is -Inf) |
| ub | Upper bound (default is Inf) |
| plb | Lower probability bound |
| pub | Upper probability bound |
| dec | Number of decimals to show |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|------------|--|
| prob_tdist | <i>Probability calculator for the t distribution</i> |
|------------|--|

Description

Probability calculator for the t distribution

Usage

```
prob_tdist(df, mean = 0, stdev = 1, lb = NA, ub = NA, plb = NA,
  pub = NA, dec = 3)
```

Arguments

| | |
|-------|-------------------------------|
| df | Degrees of freedom |
| mean | Mean |
| stdev | Standard deviation |
| lb | Lower bound (default is -Inf) |
| ub | Upper bound (default is Inf) |
| plb | Lower probability bound |
| pub | Upper probability bound |
| dec | Number of decimals to show |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-----------|--|
| prob_unif | <i>Probability calculator for the uniform distribution</i> |
|-----------|--|

Description

Probability calculator for the uniform distribution

Usage

```
prob_unif(min, max, lb = NA, ub = NA, plb = NA, pub = NA, dec = 3)
```

Arguments

| | |
|-----|----------------------------|
| min | Minimum value |
| max | Maximum value |
| lb | Lower bound |
| ub | Upper bound |
| plb | Lower probability bound |
| pub | Upper probability bound |
| dec | Number of decimals to show |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|------------|-------------------------|
| publishers | <i>Comic publishers</i> |
|------------|-------------------------|

Description

Comic publishers

Usage

```
data(publishers)
```

Format

A data frame with 3 rows and 2 variables

Details

List of comic publishers from http://stat545-ubc.github.io/bit001_dplyr-cheatsheet.html. The dataset is used to illustrate data merging / joining. Description provided in `attr(publishers,"description")`

| | |
|---------|----------------|
| radiant | <i>radiant</i> |
|---------|----------------|

Description

radiant

Launch Radiant in the default browser

Usage

```
radiant(app = c("analytics", "marketing", "quant", "base"))
```

Arguments

app Choose the app to run. One of "base", "quant", "analytics", "marketing". "analytics" is the default

Details

See <http://vnijs.github.io/radiant> for documentation and tutorials

Examples

```
if (interactive()) {
  radiant("base")
  radiant("quant")
  radiant("marketing")
  radiant("analytics")
}
```

| | |
|--------|---|
| recode | <i>Exporting the recode function from the car package</i> |
|--------|---|

Description

Exporting the recode function from the car package

| | |
|------------|------------------------------------|
| regression | <i>Linear regression using OLS</i> |
|------------|------------------------------------|

Description

Linear regression using OLS

Usage

```
regression(dataset, dep_var, indep_var, int_var = "", check = "",
  data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| dep_var | The response variable in the regression |
| indep_var | Explanatory variables in the regression |
| int_var | Interaction terms to include in the model |
| check | "standardize" to see standardized coefficient estimates. "stepwise" to apply step-wise selection of variables in estimation |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See <http://vnijs.github.io/radiant/quant/regression.html> for an example in Radiant

Value

A list of all variables used in regression as an object of class `regression`

See Also

[summary.regression](#) to summarize results
[plot.regression](#) to plot results
[predict.regression](#) to generate predictions

Examples

```
result <- regression("diamonds", "price", c("carat","clarity"))
result <- regression("diamonds", "price", c("carat","clarity"), check = "standardize")
```

| | |
|----------|--------------------------|
| repeater | <i>Repeat simulation</i> |
|----------|--------------------------|

Description

Repeat simulation

Usage

```
repeater(nr = 12, vars = "", grid = "", seed = "", name = "",
  sim = "")
```

Arguments

| | |
|------|--|
| nr | Number times to repeat the simulation |
| vars | Variables to use in repeated simulation |
| grid | Expression to use in grid search for constants |
| seed | To repeat a simulation with the same randomly generated values enter a number into Random seed input box. |
| name | To save the simulated data for further analysis specify a name in the Sim name input box. You can then investigate the simulated data by choosing the specified name from the Datasets dropdown in any of the other Data tabs. |
| sim | Return value from the simulator function |

Examples

```
result <- simulator(const = "cost 3", norm = "demand 2000 1000",
  discrete = "price 5 .3 8 .7",
  form = "profit = demand * (price - cost)")

repeater(sim = result)
```

| | |
|----------|-------------------------|
| rndnames | <i>100 random names</i> |
|----------|-------------------------|

Description

100 random names

Usage

```
data(rndnames)
```

Format

A data frame with 100 rows and 2 variables

Details

A list of 100 random names generated by listofrandomnames.com. Description provided in `attr(rndnames,"description")`

| | |
|-------------|--------------------------------|
| sample_size | <i>Sample size calculation</i> |
|-------------|--------------------------------|

Description

Sample size calculation

Usage

```
sample_size(type = "mean", err_mean = 2, sd_mean = 10, err_prop = 0.1,  
            p_prop = 0.5, conf_lev = 1.96, incidence = 1, response = 1,  
            pop_correction = "no", pop_size = 1000000)
```

Arguments

| | |
|----------------|--|
| type | Choose "mean" or "proportion" |
| err_mean | Acceptable Error for Mean |
| sd_mean | Standard deviation for Mean |
| err_prop | Acceptable Error for Proportion |
| p_prop | Initial proportion estimate for Proportion |
| conf_lev | Confidence level |
| incidence | Incidence rate (i.e., fraction of valid respondents) |
| response | Response rate |
| pop_correction | Apply correction for population size ("yes","no") |
| pop_size | Population size |

Details

See http://vnijs.github.io/radiant/quant/sample_size.html for an example in Radiant

Value

A list of variables defined in sample_size as an object of class sample_size

See Also

[summary.sample_size](#) to summarize results

Examples

```
result <- sample_size(type = "mean", err_mean = 2, sd_mean = 10)
```

sampling

*Simple random sampling***Description**

Simple random sampling

Usage

```
sampling(dataset, var, sample_size, data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| var | The variable to sample from |
| sample_size | Number of units to select |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See <http://vnijs.github.io/radiant/quant/sampling.html> for an example in Radiant

Value

A list of variables defined in sampling as an object of class `sampling`

See Also

[summary.sampling](#) to summarize results

Examples

```
result <- sampling("rndnames", "Names", 10)
```

save_factors

*Save factor scores to active dataset***Description**

Save factor scores to active dataset

Usage

```
save_factors(object)
```

Arguments

| | |
|--------|---|
| object | Return value from full_factor |
|--------|---|

Details

See http://vnijs.github.io/radiant/marketing/full_factor.html for an example in Radiant

Examples

```
result <- full_factor("diamonds", c("price", "carat", "table"))
save_factors(result)
head(diamonds)
```

| | |
|-----------------|--|
| save_membership | <i>Add a cluster membership variable to the active dataset</i> |
|-----------------|--|

Description

Add a cluster membership variable to the active dataset

Usage

```
save_membership(object)
```

Arguments

| | |
|--------|---|
| object | Return value from kmeans_clus |
|--------|---|

Details

See http://vnijs.github.io/radiant/marketing/kmeans_clus.html for an example in Radiant

See Also

[kmeans_clus](#) to generate results
[summary.kmeans_clus](#) to summarize results
[plot.kmeans_clus](#) to plot results

Examples

```
result <- kmeans_clus("shopping", vars = c("v1:v6"))
save_membership(result)
head(shopping)
```

`sdp_rm`*Standard deviation for the population na.rm = TRUE*

Description

Standard deviation for the population na.rm = TRUE

Usage

```
sdp_rm(x)
```

Arguments

x Input variable

Value

Standard deviation for the population

Examples

```
sdp_rm(rnorm(100))
```

`sd_rm`*Standard deviation with na.rm = TRUE*

Description

Standard deviation with na.rm = TRUE

Usage

```
sd_rm(x)
```

Arguments

x Input variable

Value

Standard deviation

Examples

```
sd_rm(rnorm(100))
```

| | |
|------|-----------------------|
| serr | <i>Standard error</i> |
|------|-----------------------|

Description

Standard error

Usage

```
serr(x, na.rm = TRUE)
```

Arguments

| | |
|-------|---|
| x | Input variable |
| na.rm | If TRUE missing values are removed before calculation |

Value

Standard error

Examples

```
serr(rnorm(100))
```

| | |
|-----------|---|
| set_class | <i>Alias used to set the class for analysis function return</i> |
|-----------|---|

Description

Alias used to set the class for analysis function return

Usage

```
set_class()
```

Examples

```
foo <- function(x) x^2 %>% set_class(c("foo", class(.)))
```

| | |
|----------|---------------------------|
| shopping | <i>Shopping attitudes</i> |
|----------|---------------------------|

Description

Shopping attitudes

Usage

```
data(shopping)
```

Format

A data frame with 20 rows and 7 variables

Details

Attitudinal data on shopping for 20 consumers. Description provided in `attr(shopping, "description")`

| | |
|-----------------|--|
| show_duplicated | <i>Show all rows with duplicated values (not just the first or last)</i> |
|-----------------|--|

Description

Show all rows with duplicated values (not just the first or last)

Usage

```
show_duplicated(tbl, ...)
```

Arguments

| | |
|------------------|--|
| <code>tbl</code> | Data frame to add transformed variables to |
| <code>...</code> | Variables used to evaluate row uniqueness |

Details

If an entire row is duplicated use "duplicated" to show only one of the duplicated rows. When using a subset of variables to establish uniqueness it may be of interest to show all rows that have (some) duplicate elements

Examples

```
bind_rows(mtcars, mtcars[c(1,5,7),]) %>%
  show_duplicated(mpg, cyl)
bind_rows(mtcars, mtcars[c(1,5,7),]) %>%
  show_duplicated
```

| | |
|------------------------|---|
| <code>sig_stars</code> | <i>Add stars '***' to a data.frame (from broom's 'tidy' function) based on p.values</i> |
|------------------------|---|

Description

Add stars '***' to a data.frame (from broom's 'tidy' function) based on p.values

Usage

```
sig_stars(pval)
```

Arguments

| | |
|-------------------|--------------------|
| <code>pval</code> | Vector of p-values |
|-------------------|--------------------|

Details

Add stars to output from broom's 'tidy' function

Value

A vector of stars

Examples

```
sig_stars(c(.0009, .049, .009, .4, .09))
```

| | |
|------------------------|--|
| <code>simulater</code> | <i>Simulate data for decision analysis</i> |
|------------------------|--|

Description

Simulate data for decision analysis

Usage

```
simulater(const = "", norm = "", unif = "", discrete = "", form = "",
  seed = "", name = "", nr = 1000, dat = NULL)
```

Arguments

| | |
|--------------------|--|
| <code>const</code> | A string listing the constants to include in the analysis (e.g., "cost = 3; size = 4") |
| <code>norm</code> | A string listing the normally distributed random variables to include in the analysis (e.g., "demand 2000 1000" where the first number is the mean and the second is the standard deviation) |
| <code>unif</code> | A string listing the uniformly distributed random variables to include in the analysis (e.g., "demand 0 1" where the first number is the minimum value and the second is the maximum value) |

| | |
|----------|--|
| discrete | A string listing the random variables with a discrete distribution to include in the analysis (e.g., "price 5 .3 8 .7" where for each pair of numbers the first is the value and the second the probability) |
| form | A string with the formula to evaluate (e.g., "profit = demand * (price - cost)") |
| seed | To repeat a simulation with the same randomly generated values enter a number into Random seed input box. |
| name | To save the simulated data for further analysis specify a name in the Sim name input box. You can then investigate the simulated data by choosing the specified name from the Datasets dropdown in any of the other Data tabs. |
| nr | Number of simulation runs |
| dat | Data list from previous simulation. Used by repeater function |

Details

See <http://vnijs.github.io/radiant/quant/simulator.html> for an example in Radiant

Value

A data.frame with the created variables

See Also

[summary.simulator](#) to summarize results

[plot.simulator](#) to plot results

Examples

```
result <- simulator(const = "cost 3", norm = "demand 2000 1000",
  discrete = "price 5 .3 8 .7",
  form = "profit = demand * (price - cost)")
```

single_mean

Compare a sample mean to a population mean

Description

Compare a sample mean to a population mean

Usage

```
single_mean(dataset, var, comp_value = 0, alternative = "two.sided",
  conf_lev = 0.95, dec = 3, data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| var | The variable selected for the mean comparison |
| comp_value | Population value to compare to the sample mean |
| alternative | The alternative hypothesis ("two.sided", "greater", or "less") |
| conf_lev | Span for the confidence interval |
| dec | Number of decimals to show |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/quant/single_mean.html for an example in Radiant

Value

A list of variables defined in `single_mean` as an object of class `single_mean`

See Also

`summary.single_mean` to summarize results

`plot.single_mean` to plot results

Examples

```
single_mean("diamonds", "price")
```

single_prop

Compare a sample proportion to a population proportion

Description

Compare a sample proportion to a population proportion

Usage

```
single_prop(dataset, var, lev = "", comp_value = 0.5,
  alternative = "two.sided", conf_lev = 0.95, dec = 3, data_filter = "")
```

Arguments

| | |
|-------------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| var | The variable selected for the proportion comparison |
| lev | The factor level selected for the proportion comparison |
| comp_value | Population value to compare to the sample proportion |
| alternative | The alternative hypothesis ("two.sided", "greater", or "less") |

| | |
|-------------|--|
| conf_lev | Span of the confidence interval |
| dec | Number of decimals to show |
| data_filter | Expression entered in, e.g., Data > View to filter the dataset in Radiant. The expression should be a string (e.g., "price > 10000") |

Details

See http://vnijs.github.io/radiant/quant/single_prop.html for an example in Radiant

Value

A list of variables used in single_prop as an object of class single_prop

See Also

[summary.single_prop](#) to summarize the results

[plot.single_prop](#) to plot the results

Examples

```
result <- single_prop("diamonds", "cut")
result <- single_prop("diamonds", "clarity", lev = "IF", comp_value = 0.05)
```

| | |
|------|---|
| skew | <i>Exporting the skew function from the psych package</i> |
|------|---|

Description

Exporting the skew function from the psych package

| | |
|--------|---------------------------------------|
| square | <i>Calculate square of a variable</i> |
|--------|---------------------------------------|

Description

Calculate square of a variable

Usage

```
square(x)
```

Arguments

x Input variable

Value

x^2

`ssh`*Hide warnings and messages and return invisible*

Description

Hide warnings and messages and return invisible

Usage

```
ssh(...)
```

Arguments

... Inputs to keep quiet

Details

Adapted from <http://www.onthelambda.com/2014/09/17/fun-with-rprofile-and-customizing-r-startup/>

Examples

```
ssh( library(dplyr) )
```

`sshhr`*Hide warnings and messages and return result*

Description

Hide warnings and messages and return result

Usage

```
sshhr(...)
```

Arguments

... Inputs to keep quiet

Details

Adapted from <http://www.onthelambda.com/2014/09/17/fun-with-rprofile-and-customizing-r-startup/>

Examples

```
sshhr( library(dplyr) )
```

| | |
|-------------|--------------------|
| standardize | <i>Standardize</i> |
|-------------|--------------------|

Description

Standardize

Usage

```
standardize(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

If x is a numeric variable return $\text{center}(x) / \text{mean}(x)$

| | |
|------------|--|
| state_init | <i>Set initial value for shiny input</i> |
|------------|--|

Description

Set initial value for shiny input

Usage

```
state_init(inputvar, init = "")
```

Arguments

| | |
|----------|---|
| inputvar | Name shiny input |
| init | Initial value to use if state value for input not set |

Details

Useful for radio button or checkbox

Value

value for inputvar

See Also

[state_single](#)
[state_multiple](#)
[copy_from](#)

Examples

```

r_state <- list()
state_init("test")
state_init("test",0)
r_state$test <- c("a","b")
state_init("test",0)
shiny::radioButtons("rb", label = "Button:", c("a","b"), selected = state_init("rb", "a"))
r_state$rb <- "b"
shiny::radioButtons("rb", label = "Button:", c("a","b"), selected = state_init("rb", "a"))
rm(r_state)

```

state_multiple

Set initial values for shiny input from a list of values

Description

Set initial values for shiny input from a list of values

Usage

```
state_multiple(inputvar, vals, init = character(0))
```

Arguments

| | |
|----------|---|
| inputvar | Name shiny input |
| vals | Possible values for inputvar |
| init | Initial value to use if state value for input not set |

Details

Useful for select input with multiple = TRUE and when you want to use inputs selected for another tool (e.g., pre_factor and full_factor or hier_clus and kmeans_clus in Radiant)

Value

value for inputvar

See Also

[state_init](#)
[state_single](#)
[copy_from](#)

Examples

```

r_state <- list()
state_multiple("test",1:10,1:3)
r_state$test <- 8:10
state_multiple("test",1:10,1:3)
shiny::selectInput("sim", label = "Select:", c("a","b"),
  selected = state_multiple("sim", c("a","b")), multiple = TRUE)
r_state$sim <- c("a","b")
shiny::selectInput("sim", label = "Select:", c("a","b"),
  selected = state_single("sim", c("a","b")), multiple = TRUE)

```

| | |
|--------------|--|
| state_single | <i>Set initial value for shiny input from a list of values</i> |
|--------------|--|

Description

Set initial value for shiny input from a list of values

Usage

```
state_single(inputvar, vals, init = character(0))
```

Arguments

| | |
|----------|---|
| inputvar | Name shiny input |
| vals | Possible values for inputvar |
| init | Initial value to use if state value for input not set |

Details

Useful for select input with multiple = FALSE

Value

value for inputvar

See Also

[state_init](#)
[state_multiple](#)
[copy_from](#)

Examples

```

r_state <- list()
state_single("test",1:10,1)
r_state$test <- 8
state_single("test",1:10,1)
shiny::selectInput("si", label = "Select:", c("a","b"), selected = state_single("si"))
r_state$si <- "b"
shiny::selectInput("si", label = "Select:", c("a","b"), selected = state_single("si", "b"))

```

| | |
|-----------|--|
| store_glm | <i>Store residuals or predicted values generated in the glm_reg function</i> |
|-----------|--|

Description

Store residuals or predicted values generated in the glm_reg function

Usage

```
store_glm(object, data = object$dataset, type = "residuals",
  name = paste0(type, "_glm"))
```

Arguments

| | |
|--------|---|
| object | Return value from glm_reg or predict.glm_reg |
| data | Dataset name |
| type | Residuals ("residuals") or predictions ("predictions"). For predictions the dataset name must be provided |
| name | Variable name assigned to the residuals or predicted values |

Details

See http://vnijs.github.io/radiant/quant/glm_reg.html for an example in Radiant

Examples

```
result <- glm_reg("titanic", "survived", "pclass", lev = "Yes")
store_glm(result)
```

| | |
|-----------|---|
| store_reg | <i>Store residuals or predicted values generated in the regression function</i> |
|-----------|---|

Description

Store residuals or predicted values generated in the regression function

Usage

```
store_reg(object, data = object$dataset, type = "residuals",
  name = paste0(type, "_reg"))
```

Arguments

| | |
|--------|---|
| object | Return value from regression or predict.regression |
| data | Dataset name |
| type | Residuals ("residuals") or predictions ("predictions"). For predictions the dataset name must be provided |
| name | Variable name assigned to the residuals or predicted values |

Details

See <http://vnijs.github.io/radiant/quant/regression.html> for an example in Radiant

Examples

```
result <- regression("diamonds", "price", c("carat", "clarity"))
store_reg(result)
```

summary.compare_means *Summary method for the compare_means function*

Description

Summary method for the compare_means function

Usage

```
## S3 method for class 'compare_means'
summary(object, show = FALSE, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from compare_means |
| show | Show additional output (i.e., t.value, df, and confidence interval) |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/compare_means.html for an example in Radiant

See Also

[compare_means](#) to calculate results
[plot.compare_means](#) to plot results

Examples

```
result <- compare_means("diamonds", "cut", "price")
summary(result)
result <- diamonds %>% tbl_df %>% compare_means("x", "y")
summary(result)
result <- diamonds %>% tbl_df %>% group_by(cut) %>% compare_means("x", c("x", "y"))
summary(result)
```

summary.compare_props *Summary method for the compare_props function*

Description

Summary method for the compare_props function

Usage

```
## S3 method for class 'compare_props'
summary(object, show = FALSE, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from compare_props |
| show | Show additional output (i.e., chisq.value, df, and confidence interval) |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/compare_props.html for an example in Radiant

See Also

[compare_props](#) to calculate results
[plot.compare_props](#) to plot results

Examples

```
result <- compare_props("titanic", "pclass", "survived")
summary(result)
titanic %>% compare_props("pclass", "survived") %>% summary
```

summary.conjoint *Summary method for the conjoint function*

Description

Summary method for the conjoint function

Usage

```
## S3 method for class 'conjoint'
summary(object, mc_diag = FALSE, ...)
```

Arguments

| | |
|---------|---|
| object | Return value from conjoint |
| mc_diag | Shows multicollinearity diagnostics. |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/marketing/conjoint.html> for an example in Radiant

See Also

[conjoint](#) to generate results
[plot.conjoint](#) to plot results

Examples

```
result <- conjoint("mp3", dep_var = "Rating", indep_var = "Memory:Shape")
summary(result, mc_diag = TRUE)
mp3 %>% conjoint(dep_var = "Rating", indep_var = "Memory:Shape") %>% summary(., mc_diag = TRUE)
```

summary.conjoint_profiles

Summary method for the conjoint_profiles function

Description

Summary method for the conjoint_profiles function

Usage

```
## S3 method for class 'conjoint_profiles'
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from conjoint_profiles |
| ... | further arguments passed to or from other methods. |

Details

See http://vnijs.github.io/radiant/marketing/conjoint_profiles.html for an example in Radiant

See Also

[conjoint_profiles](#) to calculate results

Examples

```
cp <- c("price = c('$10','$13','$16')", "sight = c('Staggered','Not Staggered')",
      "comfort = c('Average no cupholder','Average cupholder','Large cupholder')",
      "audio.visual = c('Small plain','Large plain','Large digital')",
      "food = c('No food','Hot dogs and popcorn','Gourmet food')")
result <- conjoint_profiles("cp")
summary(result)
rm(cp, envir = .GlobalEnv)
```

summary.correlation_ *Summary method for the correlation function*

Description

Summary method for the correlation function

Usage

```
## S3 method for class 'correlation_'
summary(object, cutoff = 0, covar = FALSE, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from correlation |
| cutoff | Show only correlations larger than the cutoff in absolute value. Default is a cutoff of 0 |
| covar | Show the covariance matrix (default is FALSE) |
| ... | further arguments passed to or from other methods. |

Details

See <http://vnijs.github.io/radiant/quant/correlation.html> for an example in Radiant

See Also

[correlation](#) to calculate results
[plot.correlation_](#) to plot results

Examples

```
result <- correlation("diamonds", c("price", "carat", "clarity"))
summary(result, cutoff = .3)
diamonds %>% correlation("price:clarity") %>% summary
```

summary.cross_tabs *Summary method for the cross_tabs function*

Description

Summary method for the cross_tabs function

Usage

```
## S3 method for class 'cross_tabs'
summary(object, check = "", ...)
```


Arguments

| | |
|--------|---|
| object | Return value from cross_tabs |
| check | Show table(s) for variables var1 and var2. "observed" for the observed frequencies table, "expected" for the expected frequencies table (i.e., frequencies that would be expected if the null hypothesis holds), "chi_sq" for the contribution to the overall chi-squared statistic for each cell (i.e., $(o - e)^2 / e$), "dev_std" for the standardized differences between the observed and expected frequencies (i.e., $(o - e) / \sqrt{e}$), and "dev_perc" for the percentage difference between the observed and expected frequencies (i.e., $(o - e) / e$) |
| ... | further arguments passed to or from other methods. |

Details

See http://vnijs.github.io/radiant/quant/cross_tabs.html for an example in Radiant

See Also

[cross_tabs](#) to calculate results
[plot.cross_tabs](#) to plot results

Examples

```
result <- cross_tabs("newspaper", "Income", "Newspaper")
summary(result, check = c("observed", "expected", "chi_sq"))
newspaper %>% cross_tabs("Income", "Newspaper") %>% summary("observed")
```

| | |
|---------------|---|
| summary.dtree | <i>Summary method for the dree function</i> |
|---------------|---|

Description

Summary method for the dree function

Usage

```
## S3 method for class 'dtree'
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from simulator |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/dtree.html> for an example in Radiant

See Also

[dtree](#) to generate the results
[plot.dtree](#) to plot results

| | |
|-----------------|--|
| summary.explore | <i>Summary method for the explore function</i> |
|-----------------|--|

Description

Summary method for the explore function

Usage

```
## S3 method for class 'explore'
summary(object, top = "fun", ...)
```

Arguments

| | |
|--------|--|
| object | Return value from explore |
| top | The variable (type) to display at the top of the table |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/base/explore.html> for an example in Radiant

See Also

[explore](#) to generate summaries

Examples

```
result <- explore("diamonds", "price:x")
summary(result)
result <- explore("diamonds", "price", byvar = "cut", fun = c("length", "skew"))
summary(result)
diamonds %>% explore("price:x") %>% summary
diamonds %>% explore("price", byvar = "cut", fun = c("length", "skew")) %>% summary
```

| | |
|---------------------|--|
| summary.full_factor | <i>Summary method for the full_factor function</i> |
|---------------------|--|

Description

Summary method for the full_factor function

Usage

```
## S3 method for class 'full_factor'
summary(object, cutoff = 0, fsort = FALSE, ...)
```

Arguments

| | |
|--------|--|
| object | Return value from full_factor |
| cutoff | Show only loadings with (absolute) values above cutoff (default = 0) |
| fsort | Sort factor loadings |
| ... | further arguments passed to or from other methods |

Details

See http://vnij.s.githu.b.io/radiant/marketing/full_factor.html for an example in Radiant

See Also

[full_factor](#) to calculate results

[plot.full_factor](#) to plot results

Examples

```
result <- full_factor("diamonds",c("price","carat","depth","table","x"))
summary(result)
summary(result, cutoff = 0, fsort = FALSE)
summary(result, cutoff = 0, fsort = TRUE)
summary(result, cutoff = .5, fsort = TRUE)
diamonds %>% full_factor(c("price","carat","depth","table","x")) %>% summary
diamonds %>% full_factor(c("price","carat","depth","table","x")) %>% summary(cutoff = .5)
```

summary.glm_reg

*Summary method for the glm_reg function***Description**

Summary method for the glm_reg function

Usage

```
## S3 method for class 'glm_reg'
summary(object, sum_check = "", conf_lev = 0.95,
        test_var = "", ...)
```

Arguments

| | |
|-----------|--|
| object | Return value from glm_reg |
| sum_check | Optional output or estimation parameters. "rsme" to show the root mean squared error. "sumsquares" to show the sum of squares table. "vif" to show multicollinearity diagnostics. "confint" to show coefficient confidence interval estimates. |
| conf_lev | Confidence level to use for coefficient and odds confidence intervals (.95 is the default) |
| test_var | Variables to evaluate in model comparison (i.e., a competing models Chi-squared test) |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/glm_reg.html for an example in Radiant

See Also

[glm_reg](#) to generate the results
[plot.glm_reg](#) to plot the results
[predict.glm_reg](#) to generate predictions
[plot.glm_predict](#) to plot prediction output

Examples

```
result <- glm_reg("titanic", "survived", "pclass", lev = "Yes")
summary(result, test_var = "pclass")
res <- glm_reg("titanic", "survived", c("pclass", "sex"), int_var="pclass:sex", lev="Yes")
summary(res, sum_check = c("vif", "confint", "odds"))
titanic %>% glm_reg("survived", c("pclass", "sex", "age"), lev = "Yes") %>% summary("vif")
```

| | |
|-------------------|--|
| summary.hier_clus | <i>Summary method for the hier_clus function</i> |
|-------------------|--|

Description

Summary method for the hier_clus function

Usage

```
## S3 method for class 'hier_clus'
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from hier_clus |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/marketing/hier_clus.html for an example in Radiant

See Also

[summary.hier_clus](#) to summarize results
[plot.hier_clus](#) to plot results

Examples

```
result <- hier_clus("shopping", vars = c("v1:v6"))
summary(result)
```

| | |
|---------------------|---------------------------------------|
| summary.kmeans_clus | <i>Summary method for kmeans_clus</i> |
|---------------------|---------------------------------------|

Description

Summary method for kmeans_clus

Usage

```
## S3 method for class 'kmeans_clus'  
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from kmeans_clus |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/marketing/kmeans_clus.html for an example in Radiant

See Also

[kmeans_clus](#) to generate results
[plot.kmeans_clus](#) to plot results
[save_membership](#) to add cluster membership to the selected dataset

Examples

```
result <- kmeans_clus("shopping", vars = c("v1:v6"))  
summary(result)  
shopping %>% kmeans_clus(vars = c("v1:v6"), nr_clus = 3) %>% summary
```

| | |
|-------------|--|
| summary.mds | <i>Summary method for the mds function</i> |
|-------------|--|

Description

Summary method for the mds function

Usage

```
## S3 method for class 'mds'  
summary(object, dec = 1, ...)
```

Arguments

| | |
|--------|--|
| object | Return value from mds |
| dec | Rounding to use for output (default = 0). +1 used for coordinates. +2 used for stress measure. Not currently accessible in Radiant |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/marketing/mds.html> for an example in Radiant

See Also

[mds](#) to calculate results
[plot.mds](#) to plot results

Examples

```
result <- mds("city", "from", "to", "distance")
summary(result)
summary(result, dec = 2)
city %>% mds("from", "to", "distance") %>% summary
```

summary.pivotr

Summary method for pivotr

Description

Summary method for pivotr

Usage

```
## S3 method for class 'pivotr'
summary(object, chi2 = FALSE, shiny = FALSE, ...)
```

Arguments

| | |
|--------|--|
| object | Return value from pivotr |
| chi2 | If TRUE calculate the chi-square statistic for the (pivot) table |
| shiny | Did the function call originate inside a shiny app |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/base/pivotr.html> for an example in Radiant

See Also

[pivotr](#) to create the pivot-table using dplyr

Examples

```
pivotr("diamonds", cvars = "cut") %>% summary
pivotr("diamonds", cvars = "cut") %>% summary
pivotr("diamonds", cvars = "cut:clarity", nvar = "price") %>% summary
```

summary.pmap

*Summary method for the pmap function***Description**

Summary method for the pmap function

Usage

```
## S3 method for class 'pmap'
summary(object, cutoff = 0, ...)
```

Arguments

| | |
|--------|--|
| object | Return value from pmap |
| cutoff | Show only loadings with (absolute) values above cutoff (default = 0) |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/marketing/pmap.html> for an example in Radiant

See Also

[pmap](#) to calculate results

[plot.pmap](#) to plot results

Examples

```
result <- pmap("computer", "brand", "high_end:business")
summary(result)
summary(result, cutoff = .3)
result <- pmap("computer", "brand", "high_end:dated", pref = c("innovative", "business"))
summary(result)
computer %>% pmap("brand", "high_end:dated", pref = c("innovative", "business")) %>%
  summary
```

| | |
|--------------------|---|
| summary.pre_factor | <i>Summary method for the pre_factor function</i> |
|--------------------|---|

Description

Summary method for the pre_factor function

Usage

```
## S3 method for class 'pre_factor'
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from pre_factor |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/marketing/pre_factor.html for an example in Radiant

See Also

[pre_factor](#) to calculate results
[plot.pre_factor](#) to plot results

Examples

```
result <- pre_factor("diamonds",c("price","carat","table"))
summary(result)
diamonds %>% pre_factor(c("price","carat","table")) %>% summary
result <- pre_factor("computer","high_end:business")
summary(result)
```

| | |
|--------------------|---|
| summary.prob_binom | <i>Summary method for the probability calculator function</i> |
|--------------------|---|

Description

Summary method for the probability calculator function

Usage

```
## S3 method for class 'prob_binom'
summary(object, type = "values", ...)
```


Arguments

| | |
|--------|---|
| object | Return value from prob_binom |
| type | Probabilities or values |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|--------------------|--|
| summary.prob_chisq | <i>Summary method for the probability calculator function (Chi-squared distribution)</i> |
|--------------------|--|

Description

Summary method for the probability calculator function (Chi-squared distribution)

Usage

```
## S3 method for class 'prob_chisq'  
summary(object, type = "values", ...)
```

Arguments

| | |
|--------|---|
| object | Return value from prob_chisq |
| type | Probabilities or values |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-------------------|--|
| summary.prob_disc | <i>Summary method for the probability calculator function (discrete)</i> |
|-------------------|--|

Description

Summary method for the probability calculator function (discrete)

Usage

```
## S3 method for class 'prob_disc'  
summary(object, type = "values", ...)
```

Arguments

| | |
|--------|---|
| object | Return value from prob_disc |
| type | Probabilities or values |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|--------------------|--|
| summary.prob_fdist | <i>Summary method for the probability calculator function (F-distribution)</i> |
|--------------------|--|

Description

Summary method for the probability calculator function (F-distribution)

Usage

```
## S3 method for class 'prob_fdist'
summary(object, type = "values", ...)
```

Arguments

| | |
|--------|---|
| object | Return value from prob_fdist |
| type | Probabilities or values |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-------------------|--|
| summary.prob_norm | <i>Summary method for the probability calculator function (normal)</i> |
|-------------------|--|

Description

Summary method for the probability calculator function (normal)

Usage

```
## S3 method for class 'prob_norm'
summary(object, type = "values", ...)
```

Arguments

| | |
|--------|---|
| object | Return value from prob_norm |
| type | Probabilities or values |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|--------------------|--|
| summary.prob_tdist | <i>Summary method for the probability calculator function (t-distribution)</i> |
|--------------------|--|

Description

Summary method for the probability calculator function (t-distribution)

Usage

```
## S3 method for class 'prob_tdist'  
summary(object, type = "values", ...)
```

Arguments

| | |
|--------|---|
| object | Return value from prob_tdist |
| type | Probabilities or values |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|-------------------|---|
| summary.prob_unif | <i>Summary method for the probability calculator function (uniform)</i> |
|-------------------|---|

Description

Summary method for the probability calculator function (uniform)

Usage

```
## S3 method for class 'prob_unif'  
summary(object, type = "values", ...)
```

Arguments

| | |
|--------|---|
| object | Return value from prob_unif |
| type | Probabilities or values |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/prob_calc.html for an example in Radiant

| | |
|--------------------|---|
| summary.regression | <i>Summary method for the regression function</i> |
|--------------------|---|

Description

Summary method for the regression function

Usage

```
## S3 method for class 'regression'
summary(object, sum_check = "", conf_lev = 0.95,
        test_var = "", ...)
```

Arguments

| | |
|-----------|---|
| object | Return value from regression |
| sum_check | Optional output or estimation parameters. "rmse" to show the root mean squared error. "sumsquares" to show the sum of squares table. "vif" to show multi-collinearity diagnostics. "confint" to show coefficient confidence interval estimates. |
| conf_lev | Confidence level used to estimate confidence intervals (.95 is the default) |
| test_var | Variables to evaluate in model comparison (i.e., a competing models F-test) |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/regression.html> for an example in Radiant

See Also

[regression](#) to generate the results
[plot.regression](#) to plot results
[predict.regression](#) to generate predictions

Examples

```
result <- regression("diamonds", "price", c("carat","clarity"))
summary(result, sum_check = c("rmse","sumsquares","vif","confint"), test_var = "clarity")
result <- regression("shopping", "v1", c("v2","v3"))
summary(result, test_var = "v2")
shopping %>% regression("v1", "v2:v6") %>% summary
```

| | |
|------------------|--------------------------------------|
| summary.repeater | <i>Summarize repeated simulation</i> |
|------------------|--------------------------------------|

Description

Summarize repeated simulation

Usage

```
## S3 method for class 'repeater'
summary(object, sum_vars = "", byvar = "",
        fun = c("sum_rm", "mean_rm", "sd_rm"), ...)
```

Arguments

| | |
|----------|---|
| object | Return value from repeater |
| sum_vars | (Numerical) variables to summaries |
| byvar | Variable(s) to group data by before summarizing |
| fun | Functions to use for summarizing |
| ... | further arguments passed to or from other methods |

| | |
|---------------------|--|
| summary.sample_size | <i>Summary method for the sample_size function</i> |
|---------------------|--|

Description

Summary method for the sample_size function

Usage

```
## S3 method for class 'sample_size'
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from sample_size |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/sample_size for an example in Radiant

See Also

[sample_size](#) to generate the results

Examples

```
result <- sample_size(type = "mean", err_mean = 2, sd_mean = 10)
summary(result)
```

| | |
|------------------|---|
| summary.sampling | <i>Summary method for the sampling function</i> |
|------------------|---|

Description

Summary method for the sampling function

Usage

```
## S3 method for class 'sampling'  
summary(object, print_sf = TRUE, ...)
```

Arguments

| | |
|----------|---|
| object | Return value from sampling |
| print_sf | Print full sampling frame. Default is TRUE |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/sampling> for an example in Radiant

See Also

[sampling](#) to generate the results

Examples

```
set.seed(1234)  
result <- sampling("rndnames", "Names", 10)  
summary(result)
```

| | |
|-------------------|--|
| summary.simulater | <i>Summary method for the simulater function</i> |
|-------------------|--|

Description

Summary method for the simulater function

Usage

```
## S3 method for class 'simulater'  
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from simulater |
| ... | further arguments passed to or from other methods |

Details

See <http://vnijs.github.io/radiant/quant/simulator.html> for an example in Radiant

See Also

[simulator](#) to generate the results

[plot.simulator](#) to plot results

Examples

```
result <- simulator(norm = "demand 2000 1000")
summary(result)
```

| | |
|---------------------|--|
| summary.single_mean | <i>Summary method for the single_mean function</i> |
|---------------------|--|

Description

Summary method for the single_mean function

Usage

```
## S3 method for class 'single_mean'
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from single_mean |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/single_mean.html for an example in Radiant

See Also

[single_mean](#) to generate the results

[plot.single_mean](#) to plot results

Examples

```
result <- single_mean("diamonds", "price")
summary(result)
diamonds %>% single_mean("price") %>% summary
```

| | |
|---------------------|--|
| summary.single_prop | <i>Summary method for the single_prop function</i> |
|---------------------|--|

Description

Summary method for the single_prop function

Usage

```
## S3 method for class 'single_prop'
summary(object, ...)
```

Arguments

| | |
|--------|---|
| object | Return value from single_prop |
| ... | further arguments passed to or from other methods |

Details

See http://vnijs.github.io/radiant/quant/single_prop.html for an example in Radiant

See Also

[single_prop](#) to generate the results
[plot.single_prop](#) to plot the results

Examples

```
result <- single_prop("diamonds", "clarity", lev = "IF", comp_value = 0.05)
summary(result)
diamonds %>% single_prop("clarity", lev = "IF", comp_value = 0.05) %>% summary
```

| | |
|--------|------------------------------|
| sum_rm | <i>Sum with na.rm = TRUE</i> |
|--------|------------------------------|

Description

Sum with na.rm = TRUE

Usage

```
sum_rm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Sum of input values

Examples

```
sum_rm(1:200)
```

| | |
|-------------|---------------------|
| superheroes | <i>Super heroes</i> |
|-------------|---------------------|

Description

Super heroes

Usage

```
data(superheroes)
```

Format

A data frame with 7 rows and 4 variables

Details

List of super heroes from http://stat545-ubc.github.io/bit001_dplyr-cheatsheet.html.

The dataset is used to illustrate data merging / joining. Description provided in attr(superheroes,"description")

| | |
|------------|--|
| test_specs | <i>Add interaction terms to list of test variables if needed</i> |
|------------|--|

Description

Add interaction terms to list of test variables if needed

Usage

```
test_specs(test_var, int_var)
```

Arguments

| | |
|----------|--|
| test_var | List of variables to use for testing for regression or glm_reg |
| int_var | Interaction terms specified |

Details

See <http://vnijs.github.io/radiant/quant/regression.html> for an example in Radiant

Value

A vector of variables names to test

Examples

```
test_specs("a", c("a:b", "b:c"))
```

| | |
|-----------|---|
| the_table | <i>Function to calculate the PW and IW table for conjoint</i> |
|-----------|---|

Description

Function to calculate the PW and IW table for conjoint

Usage

```
the_table(model, dat, indep_var)
```

Arguments

| | |
|-----------|---|
| model | Tidied model results (broom) output from conjoint passed on by summary.conjoint |
| dat | Conjoint data |
| indep_var | Independent variables used in the conjoint regression |

Details

See <http://vnijs.github.io/radiant/marketing/conjoint.html> for an example in Radiant

See Also

[conjoint](#) to generate results
[summary.conjoint](#) to summarize results
[plot.conjoint](#) to plot results

Examples

```
result <- conjoint(dataset = "mp3", dep_var = "Rating", indep_var = "Memory:Shape")  
the_table(result$model, result$dat, result$indep_var)
```

| | |
|---------|--------------------------------------|
| titanic | <i>Survival data for the Titanic</i> |
|---------|--------------------------------------|

Description

Survival data for the Titanic

Usage

```
data(titanic)
```

Format

A data frame with 1043 rows and 10 variables

Details

Survival data for the Titanic. Description provided in attr(titanic,"description")

| | |
|--------------|-------------------------|
| titanic_pred | <i>Predict survival</i> |
|--------------|-------------------------|

Description

Predict survival

Usage

```
data(titanic_pred)
```

Format

A data frame with 6 rows and 3 variables

Details

Prediction data.frame for glm_reg based on the Titanic dataset

| | |
|------------|-----------------------------|
| toothpaste | <i>Toothpaste attitudes</i> |
|------------|-----------------------------|

Description

Toothpaste attitudes

Usage

```
data(toothpaste)
```

Format

A data frame with 60 rows and 10 variables

Details

Attitudinal data on toothpaste for 60 consumers. Description provided in attr(toothpaste,"description")

| | |
|---------|---|
| varp_rm | <i>Variance for the population na.rm = TRUE</i> |
|---------|---|

Description

Variance for the population na.rm = TRUE

Usage

```
varp_rm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Variance for the population

Examples

```
varp_rm(rnorm(100))
```

| | |
|-----------|--|
| var_check | <i>Check if main effects for all interaction effects are included in the model If ':' is used to select a range _indep_var_ is updated</i> |
|-----------|--|

Description

Check if main effects for all interaction effects are included in the model If ':' is used to select a range _indep_var_ is updated

Usage

```
var_check(iv, cn, intv = "")
```

Arguments

| | |
|------|---|
| iv | List of explanatory variables provided to _regression_ or _glm_ |
| cn | Column names for all explanatory variables in _dat_ |
| intv | Interaction terms specified |

Details

See <http://vnijs.github.io/radiant/quant/regression.html> for an example in Radiant

Value

'vars' is a vector of right-hand side variables, possibly with interactions, 'iv' is the list of explanatory variables, and intv are interaction terms

Examples

```
var_check("a:d", c("a", "b", "c", "d"))
var_check(c("a", "b"), c("a", "b"), "a:c")
```

| | |
|--------|-----------------------------------|
| var_rm | <i>Variance with na.rm = TRUE</i> |
|--------|-----------------------------------|

Description

Variance with na.rm = TRUE

Usage

```
var_rm(x)
```

Arguments

| | |
|---|----------------|
| x | Input variable |
|---|----------------|

Value

Variance

Examples

```
var_rm(rnorm(100))
```

| | |
|----------|------------------|
| viewdata | <i>View data</i> |
|----------|------------------|

Description

View data

Usage

```
viewdata(dataset, vars = "", filt = "", rows = NULL, na.rm = FALSE)
```

Arguments

| | |
|---------|---|
| dataset | Name of the dataframe to change |
| vars | Variables to show (default is all) |
| filt | Filter to apply to the specified dataset. For example "price > 10000" if dataset is "diamonds" (default is "") |
| rows | Select rows in the specified dataset. For example "1:10" for the first 10 rows or "n()-10:n()" for the last 10 rows (default is NULL) |
| na.rm | Remove rows with missing values (default is FALSE) |

Details

View, search, sort, etc. your data

Examples

```
if (interactive()) {
  viewdata(mtcars)
  viewdata("mtcars")
  mtcars %>% viewdata
}
```

visualize

Visualize data using ggplot2 <http://docs.ggplot2.org/current/>

Description

Visualize data using ggplot2 <http://docs.ggplot2.org/current/>

Usage

```
visualize(dataset, xvar, yvar = "", type = "hist", facet_row = ".",
  facet_col = ".", color = "none", fill = "none", bins = 10,
  smooth = 1, sbar = "mean", check = "", axes = "", alpha = 0.5,
  data_filter = "", shiny = FALSE, custom = FALSE)
```

Arguments

| | |
|-----------|--|
| dataset | Dataset name (string). This can be a dataframe in the global environment or an element in an <code>r_data</code> list from Radiant |
| xvar | One or more variables to display along the X-axis of the plot |
| yvar | Variable to display along the Y-axis of the plot (default = "none") |
| type | Type of plot to create. One of Histogram ('hist'), Density ('density'), Scatter ('scatter'), Line ('line'), Bar ('bar'), or Box-plot ('box') |
| facet_row | Create vertically arranged subplots for each level of the selected factor variable |
| facet_col | Create horizontally arranged subplots for each level of the selected factor variable |
| color | Adds color to a scatter plot to generate a heat map. For a line plot one line is created for each group and each is assigned a different color |
| fill | Group bar, histogram, and density plots by group, each with a different color |
| bins | Number of bins used for a histogram (1 - 50) |
| smooth | Adjust the flexibility of the loess line for scatter plots |
| sbar | Plot an error bar in a scatter plot where the xvar is a factor. Options are "mean" and/or "median". Default is "mean" |
| check | Add a regression line ("line"), a loess line ("loess"), or jitter ("jitter") to a scatter plot |
| axes | Flip the axes in a plot ("flip") or apply a log transformation (base e) to the y-axis ("log_y") or the x-axis ("log_x") |

| | |
|-------------|---|
| alpha | Opacity for plot elements (0 to 1) |
| data_filter | Expression used to filter the dataset. This should be a string (e.g., "price > 10000") |
| shiny | Logical (TRUE, FALSE) to indicate if the function call originate inside a shiny app |
| custom | Logical (TRUE, FALSE) to indicate if ggplot object (or list of ggplot objects) should be returned. This option can be used to customize plots (e.g., add a title, change x and y labels, etc.). See examples and http://docs.ggplot2.org/ for options. |

Details

See <http://vnijs.github.io/radiant/base/visualize.html> for an example in Radiant

Value

Generated plots

Examples

```
visualize("diamonds", "carat", "price", type = "scatter", check = "loess")
visualize("diamonds", "price:x", type = "hist")
visualize("diamonds", "carat:x", yvar = "price", type = "scatter")
visualize(dataset = "diamonds", yvar = "price", xvar = "carat", type = "scatter", custom = TRUE) +
  ggtitle("A scatterplot") + xlab("price in $")
visualize(dataset = "diamonds", xvar = "price:carat", custom = TRUE) %>%
  {.[[1]] + ggtitle("A histogram") + xlab("price in $")}
diamonds %>% visualize(c("price", "carat", "depth"), type = "density")
```

| | |
|--------------|---|
| win_launcher | <i>Create a launcher and updater for Windows (.bat)</i> |
|--------------|---|

Description

Create a launcher and updater for Windows (.bat)

Usage

```
win_launcher(app = c("analytics", "marketing", "quant", "base"))
```

Arguments

| | |
|-----|---|
| app | App to run when the desktop icon is double-clicked ("analytics", "marketing", "quant", or "base"). Default is "analytics" |
|-----|---|

Details

On Windows a file named 'radiant.bat' and one named 'update_radiant.bat' will be put on the desktop. Double-click the file to launch the specified Radiant app or update Radiant to the latest version

Examples

```
if (interactive()) {  
  if (Sys.info()["sysname"] == "Windows") {  
    win_launcher()  
    fn <- paste0(Sys.getenv("USERPROFILE"), "/Desktop/radiant.bat")  
    if (!file.exists(fn))  
      stop("Windows launcher not created")  
    else  
      unlink(fn)  
  }  
}
```


Index

*Topic **datasets**

avengers, [14](#)
city, [16](#)
computer, [21](#)
diamonds, [27](#)
mp3, [50](#)
newspaper, [51](#)
publishers, [83](#)
rndnames, [85](#)
shopping, [91](#)
superheroes, [121](#)
titanic, [122](#)
titanic_pred, [123](#)
toothpaste, [123](#)

as_character, [6](#)
as_distance, [6](#)
as_dmy, [7](#)
as_dmy_hm, [7](#)
as_dmy_hms, [8](#)
as_duration, [8](#)
as_factor, [9](#)
as_hm, [9](#)
as_hms, [10](#)
as_integer, [10](#)
as_mdy, [11](#)
as_mdy_hm, [11](#)
as_mdy_hms, [12](#)
as_numeric, [12](#)
as_ymd, [13](#)
as_ymd_hm, [13](#)
as_ymd_hms, [14](#)
avengers, [14](#)

center, [15](#)
changedata, [15](#)
ci_label, [16](#)
ci_perc, [17](#)
city, [16](#)
clean_loadings, [17](#)
combinedata, [18](#)
compare_means, [19](#), [56](#), [101](#)
compare_props, [20](#), [57](#), [102](#)
computer, [21](#)

conjoint, [21](#), [57](#), [102](#), [103](#), [122](#)
conjoint_profiles, [22](#), [30](#), [103](#)
copy_all, [23](#)
copy_from, [23](#), [97–99](#)
correlation, [24](#), [58](#), [104](#)
cross_tabs, [25](#), [59](#), [105](#)
cv, [26](#)

decile_split, [26](#)
diamonds, [27](#)
does_vary, [27](#)
dtree, [28](#), [29](#), [59](#), [60](#), [105](#)
dtree_parser, [28](#)

explore, [29](#), [31](#), [45](#), [106](#)

factorizer, [30](#)
ff_design, [30](#)
filterdata, [31](#)
flip, [31](#)
full_factor, [32](#), [60](#), [87](#), [107](#)

getclass, [33](#)
getdata, [33](#)
getsummary, [34](#)
glm_reg, [34](#), [61](#), [62](#), [76](#), [77](#), [100](#), [107](#), [108](#)

hier_clus, [35](#), [63](#), [108](#)

inverse, [36](#)
is_empty, [37](#)
is_string, [37](#)
iterms, [38](#)

kmeans_clus, [38](#), [64](#), [88](#), [109](#)
kurtosi, [39](#)

launcher, [40](#)
lin_launcher, [40](#), [40](#)
loadcsv, [41](#)
loadcsv_url, [42](#)
loadrda, [42](#)
loadrda_url, [43](#)

mac_launcher, [40](#), [43](#)

- make_dt, 44
- make_expl, 31, 44
- make_funs, 45
- make_train, 46
- max_rm, 46
- mds, 47, 65, 110
- mean_rm, 48
- median_rm, 48
- median_split, 49
- min_rm, 49
- mode_rm, 50
- mp3, 50
- mutate_each, 51

- n_missing, 52
- newspaper, 51
- normalize, 52

- p05, 53
- p25, 53
- p75, 54
- p95, 54
- pivotr, 44, 45, 55, 65, 66, 110
- plot.compare_means, 19, 56, 101
- plot.compare_props, 20, 56, 102
- plot.conjoint, 22, 57, 103, 122
- plot.correlation_, 24, 58, 104
- plot.cross_tabs, 25, 58, 105
- plot.dtree, 28, 29, 59, 105
- plot.full_factor, 32, 60, 60, 107
- plot.glm_predict, 35, 61, 62, 77, 108
- plot.glm_reg, 35, 61, 62, 62, 77, 108
- plot.hier_clus, 36, 63, 63, 108
- plot.kmeans_clus, 39, 64, 88, 109
- plot.mds, 47, 64, 110
- plot.pivotr, 65
- plot.pmap, 66, 76, 111
- plot.pre_factor, 67, 78, 112
- plot.prob_binom, 67
- plot.prob_chisq, 68
- plot.prob_disc, 68
- plot.prob_fdist, 69
- plot.prob_norm, 69
- plot.prob_tdist, 70
- plot.prob_unif, 70
- plot.reg_predict, 72
- plot.regression, 71, 72, 78, 84, 116
- plot.repeater, 73
- plot.simulator, 73, 93, 119
- plot.single_mean, 74, 94, 119
- plot.single_prop, 75, 95, 120
- pmap, 66, 75, 111
- pre_factor, 67, 78, 112

- predict.glm_reg, 35, 61, 62, 76, 100, 108
- predict.regression, 71, 72, 77, 84, 100, 116
- print.gtable, 79
- prob_binom, 67, 79, 113
- prob_chisq, 68, 80, 113
- prob_disc, 68, 80, 113
- prob_fdist, 69, 81, 114
- prob_norm, 69, 81, 114
- prob_tdist, 70, 82, 115
- prob_unif, 70, 82, 115
- publishers, 83

- radiant, 83
- radiant-package (radiant), 83
- recode, 84
- regression, 71, 72, 77, 78, 84, 100, 116
- repeater, 73, 85, 117
- rndnames, 85

- sample_size, 86, 117
- sampling, 87, 118
- save_factors, 87
- save_membership, 39, 64, 88, 109
- sd_rm, 89
- sdp_rm, 89
- serr, 90
- set_class, 90
- shopping, 91
- show_duplicated, 91
- sig_stars, 92
- simulator, 73, 92, 105, 118, 119
- single_mean, 74, 93, 119
- single_prop, 75, 94, 120
- skew, 95
- square, 95
- sshh, 96
- sshr, 96
- standardize, 97
- state_init, 97, 98, 99
- state_multiple, 97, 98, 99
- state_single, 97, 98, 99
- store_glm, 100
- store_reg, 100
- sum_rm, 120
- summary.compare_means, 19, 56, 101
- summary.compare_props, 20, 57, 102
- summary.conjoint, 22, 57, 102, 122
- summary.conjoint_profiles, 22, 30, 103
- summary.correlation_, 24, 58, 104
- summary.cross_tabs, 25, 59, 104
- summary.dtree, 28, 29, 60, 105
- summary.explore, 29, 106
- summary.full_factor, 32, 106

summary.glm_reg, [35](#), [61](#), [77](#), [107](#)
summary.hier_clus, [36](#), [63](#), [108](#), [108](#)
summary.kmeans_clus, [39](#), [64](#), [88](#), [109](#)
summary.mds, [47](#), [65](#), [109](#)
summary.pivotr, [44](#), [45](#), [66](#), [110](#)
summary.pmap, [66](#), [76](#), [111](#)
summary.pre_factor, [67](#), [78](#), [112](#)
summary.prob_binom, [112](#)
summary.prob_chisq, [113](#)
summary.prob_disc, [113](#)
summary.prob_fdist, [114](#)
summary.prob_norm, [114](#)
summary.prob_tdist, [115](#)
summary.prob_unif, [115](#)
summary.regression, [71](#), [72](#), [78](#), [84](#), [116](#)
summary.repeater, [117](#)
summary.sample_size, [86](#), [117](#)
summary.sampling, [87](#), [118](#)
summary.simulator, [93](#), [118](#)
summary.single_mean, [74](#), [94](#), [119](#)
summary.single_prop, [75](#), [95](#), [120](#)
superheroes, [121](#)

test_specs, [121](#)
the_table, [122](#)
titanic, [122](#)
titanic_pred, [123](#)
toothpaste, [123](#)

var_check, [124](#)
var_rm, [125](#)
varp_rm, [124](#)
viewdata, [125](#)
visualize, [126](#)

win_launcher, [40](#), [127](#)