Package ‘regrrr’

July 21, 2021

Type Package
Title Toolkit for Compiling, (Post-Hoc) Testing, and Plotting Regression Results
Version 0.1.2
Description Compiling regression results into a publishable format, conducting post-hoc hypothesis testing, and plotting moderating effects (the effect of X on Y becomes stronger/weaker as Z increases).
Depends R (>= 3.5.0)
Maintainer Rui K. Yang <rkzyang@gmail.com>
License GPL-3
Encoding UTF-8
Suggests testthat
RoxygenNote 7.0.2
Imports stats, purrr, dplyr, magrittr, tidyr, usdm, scales, robustbase, stringr, MuMIn, ggplot2, lspline
BugReports https://github.com/RkzYang/regrrr/issues
NeedsCompilation no
Author Rui K. Yang [aut, cre],
Luyao Peng [aut]
Repository CRAN
Date/Publication 2021-07-21 06:50:02 UTC

R topics documented:

add.n.r ................................................................. 2
add.pr ............................................................... 2
add.sig ............................................................... 3
check_cor .......................................................... 3
check_na_in ......................................................... 4
check_vif .......................................................... 4
**add.n.r**

*Add row numbers to regression result data.frame*

**Description**

Add row numbers to regression result data.frame

**Usage**

```r
add.n.r(df)
```

**Arguments**

df  
a data.frame of regression result

---

**add.pr**

*Add approximate p-value based on t score or z score, when sample size is large*

**Description**

Add approximate p-value based on t score or z score, when sample size is large

**Usage**

```r
add.pr(df, z.col = 3, p.already = FALSE)
```

**Arguments**

df  
a data.frame of regression result

z.col  
the column number of t score or z score

p.already  
whether the regression result already contains p.value
add.sig

Add significance level marks to the regression result

Description
Add significance level marks to the regression result

Usage
add.sig(df, Pr.col = 5)

Arguments
df a data.frame of regression result, e.g. summary(a_lm_model)$coefficients
Pr.col the column number of p.value

check_cor
quickly check correlation matrix, or the correlation between a particular X and all other vars could be useful for looking for relevant instrument

Description
quickly check correlation matrix, or the correlation between a particular X and all other vars could be useful for looking for relevant instrument

Usage
check_cor(data, var_name_select = NULL, d = 3)

Arguments
data a data.frame used in regression model
var_name_select to specify the variable names to be included in the table, default is NULL—all variables are included
d number of digits retained after the decimal point

Examples
data(mtcars)
check_cor(mtcars)
**check_na_in**

*quickly check the proportion of NAs in each columns of a dataframe*

**Description**
quickly check the proportion of NAs in each columns of a dataframe

**Usage**

```r
check_na_in(data, true_total = FALSE)
```

**Arguments**

- `data`: a data.frame
- `true_total`: FALSE to show the percentage, TRUE to show the true number of missing values

**Examples**

```r
data(mtcars)
check_na_in(mtcars)
```

---

**check_vif**

*quickly check the vifs in a regression model; for checking multi-collinearity*

**Description**
quickly check the vifs in a regression model; for checking multi-collinearity

**Usage**

```r
check_vif(data)
```

**Arguments**

- `data`: a data.frame used in regression model

**Examples**

```r
data(mtcars)
model <- lm(mpg ~ vs + carb + hp + wt + wt * hp, data = mtcars)
check_vif(data = model$model)
```
**Description**

Combine regression results from different models by columns

**Usage**

```r
combine_long_tab(
    tbl_1,
    tbl_2,
    tbl_3 = NULL,
    tbl_4 = NULL,
    tbl_5 = NULL,
    tbl_6 = NULL,
    tbl_7 = NULL,
    tbl_8 = NULL,
    tbl_9 = NULL,
    tbl_10 = NULL,
    tbl_11 = NULL,
    tbl_12 = NULL,
    tbl_13 = NULL,
    tbl_14 = NULL,
    tbl_15 = NULL,
    tbl_16 = NULL,
    tbl_17 = NULL,
    tbl_18 = NULL,
    tbl_19 = NULL,
    tbl_20 = NULL
)
```

**Arguments**

- `tbl_1`: the 1st data.frame of regression result
- `tbl_2`: the 2nd data.frame of regression result
- `tbl_3`: the 3rd data.frame of regression result
- `tbl_4`: the 4th data.frame of regression result
- `tbl_5`: the 5th data.frame of regression result
- `tbl_6`: the 6th data.frame of regression result
- `tbl_7`: the 7th data.frame of regression result
- `tbl_8`: the 8th data.frame of regression result
- `tbl_9`: the 9th data.frame of regression result
- `tbl_10`: the 10th data.frame of regression result
Examples

data(mtcars)
m1 <- lm(mpg ~ vs + carb + hp + wt + wt * hp, data = mtcars)
m2 <- update(m1, . ~ . + wt * vs)
summary(m1)
summary(m2)
combine_long_tab(to_long_tab(summary(m1)$coef),
    to_long_tab(summary(m2)$coef))

---

**compare_models**

Compare regression models, which is compatible with the reg.table output # updated 9/13/2018 #

**Description**

Compare regression models, which is compatible with the reg.table output # updated 9/13/2018 #

**Usage**

```r
compare_models(model1,
    model2,
    model3 = NULL,
    model4 = NULL,
    model5 = NULL,
    model6 = NULL,
    model7 = NULL,
    model8 = NULL,
    model9 = NULL,
    model10 = NULL,
    model11 = NULL,
    model12 = NULL,
    model13 = NULL,
    model14 = NULL,
    model15 = NULL,
    model16 = NULL,
    model17 = NULL,
    model18 = NULL,
    model19 = NULL,
    model20 = NULL)
```
model13 = NULL,
model14 = NULL,
model15 = NULL,
model16 = NULL,
model17 = NULL,
model18 = NULL,
model19 = NULL,
model20 = NULL,
likelihood.only = FALSE,
round.digit = 3,
main.effect.only = NULL,
intn.effect.only = NULL
)

Arguments
model1 the 1st regression model
model2 the 2nd regression model
model3 the 3rd regression model
model4 the 4th regression model
model5 the 5th regression model
model6 the 6th regression model
model7 the 7th regression model
model8 the 8th regression model
model9 the 9th regression model
model10 the 10th regression model
model11 the 11th regression model
model12 the 12th regression model
model13 the 13th regression model
model14 the 14th regression model
model15 the 15th regression model
model16 the 16th regression model
model17 the 17th regression model
model18 the 18th regression model
model19 the 19th regression model
model20 the 20th regression model
likelihood.only whether or not to output the likelihood
round.digit number of decimal places to retain
main.effect.only specify col number of alternative main-effect models, if any
intn.effect.only specify col number of alternative moderator models, if any
cor.table

**Description**

make the correlation matrix from the data.frame used in regression

**Usage**

```r
cor.table(
  data,
  data_to_combine = NULL,
  var_name_select = NULL,
  all.var.names = NULL,
  d = 2
)
```

**Arguments**

- `data` a data.frame used in regression model, e.g. `model$model`
- `data_to_combine` another data.frame used for regression model, e.g. when you have similar set of X’s but different Y’s
- `var_name_select` optional: to specify the variable names used in regression to be included in the correlation matrix
- `all.var.names` optional: to rename all variable names, a string vector
- `d` number of decimal places to retain

**Examples**

```r
data(mtcars)
model <- lm(mpg ~ vs + carb + hp + wt + wt * hp , data = mtcars)
cor.table(data = model$model)
```
### load.pkgs

**Description**
load multiple packages

**Usage**

```r
load.pkgs(pkg_name_vec)
```

**Arguments**

- `pkg_name_vec`: a string vector of package names

**Examples**

```r
## Not run:
load.pkgs(c("dplyr", "car", "purrr"))

## End(Not run)
```

---

### plot_effect

**Description**
plotting the marginal effect of X on Y, with or without one or multiple interaction terms

**Usage**

```r
plot_effect(
    reg.coef,
    data,
    model,
    by_color = FALSE,
    x_var.name = NULL,
    y_var.name = NULL,
    moderator.name = NULL,
    min_x = 0.001,
    max_x = 0.999,
    mdrt_quantile_05 = NULL,
    mdrt_quantile_50 = NULL,
    mdrt_quantile_95 = NULL,
)```
mod.n.sd = 1,
confidence_interval = FALSE,
v = NULL,
CI_Ribbon = FALSE,
title = NULL,
xlab = "X_Var.name",
ylab = "Y_Var.name",
moderator.lab = "Moderator_name",
mdrt.low.name = "Low",
mdrt.mid.name = NULL,
mdrt.high.name = "High",
y.high.lim = NULL,
y.low.lim = NULL,
spline_labels = c("LHS", "RHS")
)

Arguments

reg.coef a coefficient matrix of regression result, e.g. summary(lm_model)$coef
data the data used in regression, a data frame
model the model object, such as a "lm" object
by_color plot interactions by colors, otherwise by line types
x_var.name x name in the regression model, a string
y_var.name y name in the regression model, a string
moderator.name moderating variable name in the regression model, a string
min_x the min of x scale, in percentile of x
max_x the max of x scale, in percentile of x
mdrt_quantile_05 set the low level of moderator, in percentile
mdrt_quantile_50 set the middle level of moderator, in percentile
mdrt_quantile_95 set the high level of moderator, in percentile
mod.n.sd set the moderating strength, in the number of s.d. units, which can take negative values
confidence_interval if TRUE, plot confidence intervals
v a customized variance-covariance matrix
CI_Ribbon if TRUE, plot confidence interval ribbons, if FALSE, plot error bars
title the title of the plot
xlab label of X
ylab label of Y
moderator.lab label of moderator
plot_effect

**Examples**

```r
## Not run:
data(mtcars)
m1 <- lm(mpg ~ vs + carb + hp + wt + wt * hp , data = mtcars)
plot_effect(reg.coef = summary(m1)$coefficients,
data = mtcars, model = m1,
x_var.name = "wt", y_var.name = "mpg", moderator.name = "hp",
confidence_interval = TRUE, CI_Ribbon = TRUE,
xlab = "Weight", ylab = "MPG", moderator.lab = "Horsepower")
## End(Not run)

# @examples

## Not run:
data(mtcars)
m2 <- lm(mpg ~ vs + carb + hp + wt + wt * hp + wt * vs, data = mtcars)
plot_effect(reg.coef = summary(m2)$coefficients,
data = mtcars, model = m2,
x_var.name = "wt", y_var.name = "mpg", moderator.name = "hp",
confidence_interval = TRUE, CI_Ribbon = FALSE,
xlab = "Weight", ylab = "MPG", moderator.lab = "Horsepower")
## End(Not run)

# @examples

## Not run:
data(mtcars)
m3 <- lm(mpg ~ vs + carb + hp + lspline(wt, knots = 4, marginal = FALSE) * hp, data = mtcars)
plot_effect(reg.coef = summary(m3)$coefficients,
data = mtcars, model = m3, x_var.name = "wt", y_var.name = "mpg", moderator.name = "hp",
xlab="Weight", ylab="MPG", moderator.lab="Horsepower")
## End(Not run)

## Not run:
# this shows the function is compatible with ggplot2 customization
library(extrafont)
m1 <- lm(mpg ~ vs + carb + hp + wt + wt * hp , data = mtcars)
plot_effect(reg.coef = summary(m1)$coefficients,
data = mtcars, model = m1,
x_var.name = "wt", y_var.name = "mpg", moderator.name = "hp",
confidence_interval = TRUE, CI_Ribbon = TRUE,
CI_Ribbon = TRUE,
```

---

**mdrt.low.name** the label of low-level moderator

**mdrt.mid.name** the label of mid-level moderator

**mdrt.high.name** the label of high-level moderator

**y.high.lim** specify the upper limit of y

**y.low.lim** specify the lower limit of y

**spline_labels** label of the spline variable; when the main variable is a linear spline and spline labels are supplied, the moderation effect will be presented by facets.
scale_01

scale_01(x)

Arguments

x a vector
**test_coef_equality**

*testing equality of two coefficients (difference between coefficients of regressors), a Wald* note: if \( v \) is not alternatively specified, use `car::linearHypothesis(lm_model, "X1 = X2")`

**Description**

testing equality of two coefficients (difference between coefficients of regressors), a Wald test note: if \( v \) is not alternatively specified, use `car::linearHypothesis(lm_model, "X1 = X2")`

**Usage**

test_coef_equality(model, var1.name, var2.name, v = NULL)

**Arguments**

- **model**: the model object, such as a "lm" object
- **var1.name**: X1 name in model, a string
- **var2.name**: X2 name in model, a string
- **v**: a customized variance-covariance matrix

```r
data(mtcars) m1 <- lm(mpg ~ vs + carb + hp + wt + wt * hp , data = mtcars)
summary(m1) test_coef_equality(model = m1, var1.name = "carb", var2.name = "hp")
```

**test_tilted_slopes**

*significance of regression slope (the marginal effect) under moderation testing restriction: the sig. of beta_x under the moderation of z1, with or without additional interaction terms (z2, z3, etc.)*

**Description**

significance of regression slope (the marginal effect) under moderation testing restriction: the sig. of beta_x under the moderation of z1, with or without additional interaction terms (z2, z3, etc.)

**Usage**

test_tilted_slopes(
  reg.coef,
  v = NULL,
  model,
  x.var.name,
  moderator.name,
  mod.n.sd = 1,
  data,
  t.value.col = 3,
  Pr.col = 4
)
```
Arguments

`reg.coef`  a data.frame (or matrix) of regression result or a coeftest object, e.g. `summary(lm_model)$coef, coeftest(lm_model, cluster.vcov(lm_model, cbind(data$group1, data$group2)))`

`v`  a customized variance-covariance matrix

`model`  the model object, such as a "lm" object

`x_var.name`  main independent variable name in model, a string

`moderator.name`  moderator name in model, a string

`mod.n.sd`  specify the strength of the moderating effects, in the unit of s.d.s of the moderator, which can take negative values

`data`  data used for regression

`t.value.col`  col number of the t-score in reg.coef

`Pr.col`  col number of the Prob.(>|t|)) in reg.coef

Examples

```r
data(mtcars)
m1 <- lm(mpg ~ vs + carb + hp + wt + wt * hp, data = mtcars)
test_tilted_slopes(reg.coef = summary(m1)$coef, model = m1,
x_var.name = "wt", moderator.name = "hp", data = mtcars)
```

Description

Convert the regression result to the long format: the standard errors are in parentheses and beneath the betas

Usage

```r
to_long_tab(reg.coef, d = 3, t.value.col = 3, Pr.col = 4)
```

Arguments

`reg.coef`  a data.frame (or matrix) of regression result or a coeftest object, e.g. `summary(lm_model)$coef, coeftest(lm_model, cluster.vcov(lm_model, cbind(data$group1, data$group2)))`

`d`  number of decimal places to retain

`t.value.col`  col number of the t-score in the reg.coef data.frame

`Pr.col`  col number of the Prob.(>|t|)) in the reg.coef data.frame
to_long_tab

Examples

data(mtcars)
ml <- lm(mpg ~ vs + carb + hp + wt + wt * hp, data = mtcars)
to_long_tab(reg.coef = summary(ml)$coef)
Index

add.n.r, 2
add.pr, 2
add.sig, 3
check_cor, 3
check_na_in, 4
check_vif, 4
combine_long_tab, 5
compare_models, 6
cor.table, 8
load.pkgs, 9
plot_effect, 9
regrrr, 12
regrrr-package (regrrr), 12
scale_01, 12
test_coef_equality, 13
test_tilted_slopes, 13
to_long_tab, 14