Package ‘RegrCoeffsExplorer’

February 2, 2025

Title Efficient Visualization of Regression Coefficients for Lm(), Glm(), and Glmnet() Objects

Version 1.2.0

Imports dplyr, ggpubr, gridExtra, glmnet, ggplot2, magrittr, grDevices, rlang, stats

Description The visualization tool offers a nuanced understanding of regression dynamics, going beyond traditional per-unit interpretation of continuous variables versus categorical ones. It highlights the impact of unit changes as well as larger shifts like interquartile changes, acknowledging the distribution of empirical data. Furthermore, it generates visualizations depicting alterations in Odds Ratios for predictors across minimum, first quartile, median, third quartile, and maximum values, aiding in comprehending predictor-outcome interplay within empirical data distributions, particularly in logistic regression frameworks.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Suggests faraway, knitr, MASS, psych, rmarkdown, selectiveInference, survival

URL `https://vadimtyuryaev.github.io/RegrCoeffsExplorer/`

NeedsCompilation no

Author Vadim Tyuryaev [aut, cre] (<`https://orcid.org/0009-0008-1361-6265`>),
Aleksandr Tsybakin [aut],
Jane Heffernan [aut],
Hanna Jankowski [aut],
Kevin McGregor [aut]

Maintainer Vadim Tyuryaev <`vadim.tyuryaev@gmail.com`>

Repository CRAN

Date/Publication 2025-02-01 23:40:10 UTC

Contents

`detransform` *Detransform Centered/Scaled Data*

Description

This function back transforms centered/scaled data.

Usage

```
detransform(x_data, ...)
```

Arguments

`x_data`	Model matrix that has been centered and/or scaled.
`...`	Additional arguments specifying centering/scaling attributes.

Details

The following additional arguments can be passed:

- `attr_center` : Centering attributes.
  - If none specified, `attr(x_data,'scaled:center')` is utilized.
- `attr_scale` : Scaling attributes.
  - If none specified, `attr(x_data,'scaled:scale')` is utilized.

Value

Returns de-centered and de-scaled model matrix.

See Also

- `scale` for centering and scaling.
- `all.equal` for testing "near equality".

Examples

```
# Set seed for reproducibility
set.seed(1964)
# Generate a 10x10 matrix with random numbers
original_data <- matrix(rnorm(100), nrow = 10)
# Scale and center the data
scaled_centered_data <- scale(original_data, center = TRUE,
scale = TRUE)
# Transform the scaled/centered data back to its original form
original_data_recovered <- detransform(scaled_centered_data)
# Compare the original data and the recovered data
all.equal(original_data,original_data_recovered)
```

`plot_OR` *Plot Odds Ratio*

Description

The function accepts input in the form of a generalized linear model (GLM) or a glmnet object, specifically those employing binomial families, and proceeds to generate a suite of visualizations illustrating alterations in Odds Ratios for given predictor variable corresponding to changes between minimum, first quartile (Q1), median (Q2), third quartile (Q3), and maximum values observed in empirical data. These plots offer a graphical depiction of the influence exerted by individual predictors on the odds of the outcome, facilitating a clear interpretation of their respective significance. Such a tool aids in comprehending the interplay between predictors and outcomes within the logistic regression framework, particularly within the context of empirical data distributions.

Usage

```
plot_OR(
  func,
  data,
  var_name,
  color_filling = grey.colors(4, start = 0.1, end = 0.9),
  verbose = FALSE
)
```

Arguments

`func`	A fitted model object with binomial family, expected to be one of the following classes: • `glm lm` : Generalized Linear Models. • `lognet glmnet` : Regularized Logistic Models.
`data`	Input data frame that was used to fit the input function (`data.frame` object).
`var_name`	Name of a variable to plot graphs for (`string` object).
`color_filling`	Vector with color numbers to plot in bar plot (`vector` object). Default is `grey.colors(4, start=0.1, end=0.9)`.
`verbose`	`TRUE` to print additional information such as Warnings, `FALSE` otherwise (`bool` object). Default is `FALSE`.

Value

A list with the following components:

- `$BarPlot` : A `ggplot` object that visualizes dependency of a change in Variable values on function's Odds Ratio values.
- `$BoxPlot` : A `ggplot` object that visualizes distribution of data points for a given variable.
- `$SidebySide` : A `ggarrange` object containing both visualizations side-by-side.

Examples

```
### Prepare Sample Binomial Data
set.seed(42)
obs_num = 100

x1 = rnorm(obs_num)
x2 = rnorm(obs_num)
x3 = rnorm(obs_num)

prob = plogis(1 + 0.3 * x1 + 0.2 * x2 + 0.1 * x3)
y = rbinom(obs_num, 1, prob)
data = data.frame(x1, x2, x3, y)


### GLM Object Exmaple
# Get GLM model
glm_object = glm(y ~ x1 + x2 + x3,
                 family=binomial(link="logit"),
                 data=data)
summary(glm_object)

# Plot Odds Ratio graphs
plot_OR(glm_object, data, var_name="x2")$"SidebySide"


### GLMNET Object Example
require(glmnet)

# Get Lasso model
y_lasso = data$y
x_lasso = model.matrix(as.formula(paste("~",
                                        paste(colnames(subset(data,
                                                              select=-c(y))),
                                              collapse = "+"),
                                        sep = "")),
                       data=data)
x_lasso = x_lasso[,-1]
ndim_lasso = dim(x_lasso)[1]

# Select the 1se lambda from cross validation
cv_model_lasso = cv.glmnet(x_lasso, y_lasso, family="binomial", alpha=1)
lambda_lasso = cv_model_lasso$lambda.1se
plot(cv_model_lasso)

# Get a model with the specified lambda
model_lasso = glmnet(x_lasso, y_lasso, family="binomial",
                     alpha=0.5, lambda=lambda_lasso)
summary(model_lasso)

# Plot Odds Ratio graphs
plot_OR(model_lasso, data, var_name="x2")$"SidebySide"
```

vis_reg — *Visualize Regression Coefficients Within the Context of Empirical Data*

Description

Typically, regression coefficients for continuous variables are interpreted on a per-unit basis and compared against coefficients for categorical variables. However, this method of interpretation is flawed as it overlooks the distribution of empirical data. This visualization tool provides a more nuanced understanding of the regression model's dynamics, illustrating not only the immediate effect of a unit change but also the broader implications of larger shifts such as interquartile changes.

Usage

```
vis_reg(object, ...)
```

Arguments

`object`	A fitted model object, expected to be one of the following classes: • `lm` : Linear Models. • `glm lm` : Generalized Linear Models. • `elnet glmnet` : Regularized Linear Models. • `lognet glmnet` : Regularized Logistic Models. • `fixedLassoInf` : Inference for the lassso for the linear models. • `fixedLogitLassoInf` : Inference for the lassso for the logistic models.
`...`	Additional parameters.Please refer to details.

Details

The following additional arguments can be passed:

- `CI`: A logical value indicating whether to include Confidence Intervals.
  - The default is `FALSE`.
  - For `fixedLassoInf` or `fixedLogitLassoInf` classes it is set to `TRUE`.
  - `confint()` is used to generate CIs for the `lm` and `glm lm` classes.
  - `If CIs are desired for the regularized models, please, fit your model using` fixed-LassoInf()`function from the`selectiveInference`package following the steps outlined in the documentat`
- `x_data_orig`: Original non-centered and non-scaled model matrix without intercept.
  - Please, pass the model matrix when CIs desired for `fixedLassoInf` and/or `fixedLogitLassoInf` object classes with penalty factors.
  - For objects fitted without penalty factors this argument is not required as original data can be reconstructed from the object passed.
- `intercept`: A logical value indicating whether to include the intercept.
  - The default is `FALSE`.
  - For the regularized models it is set to `FALSE`.

- `title` : Custom vectors of strings specifying titles for both plots.
- `alpha` : A numeric value between 0 and 1 specifying the significance level.
  - The default is 0.05.
- `palette` : Custom vector of colors to highlight the direction of estimated regression coefficients or Odds Ratio.
  - Grey scale is implemented by default.
  - Values at low and high ends of the grey scale palette can be specified.
- `start` : grey value at low end of palette.
  - The default value is 0.5.
- `end` : grey value at high end of palette.
  - The default value is 0.9.
- `eff_size_diff` : A vector specifying which values to utilize for realized effect size calculation.It is applied to all independent variables. By default it is c(4,2) which is Q3 - Q1. The following coding scheme is used:
  - 1 is the minimum.
  - 2 is the first quartile.
  - 3 is the second quartile.
  - 4 is the third quartile.
  - 5 is the maximum.
- `round_func` : A string specifying how to round the realized effect size.
  - Can be either "floor", "ceiling", or "none".
  - The default value is "none".
- `glmnet_fct_var` : names of categorical variables for regularized models.
  - Glmnet treats all variables as numeric.
  - If any of the variables utilized are, in fact, categorical, please, specify their name(s).
  - Please, note that that by default `model.matrix()`will create k-1 dummy variables in lieu of k levels of a categorical variable. For example,if you have a factor variable called "sex" with two levels 0 and 1, and 0 being the base level, `mode.matrix()` will create a dummy variable called "sex1". Please, utilize the names created by `mode.matrix()` here and not the original factor name.
- `verbose` : A logical value indicating whether to display warning messages.
  - The default is `FALSE`.

Please note the following:

- Only `Gaussian` and `binomial` families are currently supported.
- Certain steps should be followed in order to produce Confidence Intervals for the regularized models. Please, refer to the vignette for the `vis_reg()` function and the documentation of the `selectiveInference` package.
- Penalty factor of 0 is not currently supported and no Confidence Intervals will be produced in this case.

Value

A list with the following components:

- `$PerUnitVis`: A `ggplot` object that visualizes regression coefficients on a per-unit basis
- `$RealizedEffectVis`: A `ggplot` object that visualizes regression coefficients on a basis of realized effect calculation.
- `$SidebySide`: A `grob` object containing both visualizations side-by-side.

See Also

- `lm` for linear models.
- `glm` for generalized linear models.
- `glmnet` and `cv.glmnet` for lasso and elastic-net regularized generalized linear models.
- `model.matrix` for design matrices.
- `ggplot` for ggplot objects.
- `arrangeGrob` for grobs, gtables, and ggplots.
- `fixedLassoInf` for post-selection inference.

Examples

```
# Set seed for reproducibility
set.seed(38)
# Set the number of observations
n = 1000
# Generate predictor variables
X1 = rnorm(n)
X2 = rnorm(n)
X3 = rnorm(n)
# Define coefficients for each predictor
beta_0 = -1
beta_1 = 0.5
beta_2 = -0.25
beta_3 = 0.75
# Generate the latent variable
latent_variable = beta_0 + beta_1 * X1+ beta_2 * X2 + beta_3 * X3
# convert it to probabilities
p = pnorm(latent_variable)
# Generate binomial outcomes based on these probabilities
y = rbinom(n, size = 1, prob = p)
# Fit a GLM with a probit link
glm_model <- glm(y ~ X1 + X2 + X3, family = binomial(link = "probit"),
                 data = data.frame(y, X1, X2, X3))
# Specify additional parameters and Plot Odds Ratio for the Realized Effect
vis_reg(glm_model, CI=TRUE,intercept=TRUE,
        palette=c("greenyellow","red4"))$RealizedEffectVis
```

Index