Package: nRegression (via r-universe)

September 17, 2024

Title Simulation-Based Calculations of Sample Size for Linear and Logistic Regression

Version 0.5.1

Depends R (>= 4.0.0)

Description Provides a function designed to estimate the minimal sample size required to attain a specific statistical power in the context of linear regression and logistic regression models through simulations.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, dplyr, testthat (>= 3.0.0), devtools

Imports data.table, covr, simitation, stats

VignetteBuilder knitr

NeedsCompilation no

Author David Shilane [aut], Srivastav Budugutta [ctb, cre], Mayur Bansal [ctb]

Maintainer Srivastav Budugutta <sb4788@columbia.edu>

Config/testthat/edition 3

Date/Publication 2023-10-17 22:30:07 UTC

Repository https://srivastavbudugutta.r-universe.dev

RemoteUrl https://github.com/cran/nRegression

RemoteRef HEAD

RemoteSha 7bea826310f862a4cfa17289f57b2d5fd64ab2d4

Contents

nRegression *nRegression*

Description

The nRegression package was designed to estimate the minimal sample size required to attain a specific statistical power in the context of linear regression and logistic regression models through simulations.

Usage

```
nRegression(
  the.steps,
  num.experiments,
  model.type = "lm",
  the.formula,
  the.variable,
  seed,
  n.start = 50,
  n.min = 1,
  n.max = 10000,
  increment = 50,
  stop.threshold = 5,
  power = 0.8,
  conf.level = 0.95,
  verbose = TRUE,
  vstr = 3.6
)
```

Arguments

the.steps	The steps are ordered to describe which variables will be calculated. Note that the variables with a dependence on other variables must be specified after all of its inputs.
num.experiments	The number of experiments to perform at each sample size to estimate the power.
model.type	The type of regression model to fit.By default it is linear regression
the.formula	The formula for the regression model to fit.
the.variable	the variable to test
seed	This parameter is used to set a specific random seed, ensuring that the results are reproducible if the same seed is used.
n.start	The initial sample size to start the simulations. By default it is 50.
n.min	The minimum possible sample size. By default it is 1
n.max	The maximum possible sample size. By default it is 10000

increment	The increment to increase the sample size at each iteration. By default it is 50.
stop.threshold	The number of iterations to stop after if the desired power is not achieved.
power	describes the statistical power
conf.level	The confidence level for the statistical power.
verbose	Logical. If TRUE, the function will print information about each iteration.
vstr	Variance inflation factor. Used in the simulation of the response variable.

Value

A list containing the following elements:

- 'min.sample.size': The estimated minimum sample size to achieve the desired power.
- 'power': The statistical power achieved with the estimated sample size.
- 'iterations': A data frame with details about each iteration of the simulation. Each row represents an iteration and contains the sample size, the estimated power, and the upper and lower bounds for the sample size for that iteration.

Examples

```
require(data.table)
require(simitation)
power = 0.9
step.age <- "Age ~ N(45, 10)"
step.female <- "Female ~ binary(0.53)"
step.health.percentile <- "Health.Percentile ~ U(0,100)"
step.exercise.sessions <- "Exercise.Sessions ~ Poisson(2)"
step.diet <- "Diet ~ sample(('Light', 'Moderate', 'Heavy'), (0.2, 0.45, 0.35))"
step.healthy.lifestyle <- "Healthy.Lifestyle ~ logistic(log(0.45) - 0.1 *
(Age -45) + 0.05 * Female + 0.01 * Health.Percentile + 0.5 *
Exercise.Sessions - 0.1 * (Diet == 'Moderate') - 0.4 * (Diet == 'Heavy'))"
step.weight <- "Weight ~ lm(150 - 15 * Female + 0.5 * Age - 0.1 *
Health.Percentile - 0.2 * Exercise.Sessions  + 5 * (Diet == 'Moderate') + 15 *
(Diet == 'Heavy') - 2 * Healthy.Lifestyle + N(0, 10))"
the.steps <- c(step.age, step.female, step.health.percentile, step.exercise.sessions,
 step.diet, step.healthy.lifestyle, step.weight)
the.formula.logistic <- Healthy.Lifestyle ~ Age + Female + Health.Percentile +
 Exercise.Sessions + Weight
the.variable = "Exercise.Sessions"
conf.level = 0.95
model.type = "logistic"
seed = 41
vstr = 3.6
num.experiments = 10
n.start = 200
n.min = 1
n.max = 300
increment = 100
stop.threshold = 1
n.logistic = nRegression(the.steps = the.steps, num.experiments = num.experiments,
 the.formula = the.formula.logistic, the.variable = the.variable,
```

```
seed = seed, n.start = n.start, n.min = n.min, n.max = n.max,
increment = increment, stop.threshold = stop.threshold, power = power,
model.type = model.type, verbose = TRUE)
```

Index