Package ‘sclr’

November 26, 2019

Title  Scaled Logistic Regression
Version  0.3.0
Description  Maximum likelihood estimation of the scaled logit model parameters
License  MIT + file LICENSE
Encoding  UTF-8
LazyData  true
RoxygenNote  7.0.0
Depends  R (>= 3.6.0)
Imports  broom, tibble, dplyr, rlang, stats
Suggests  knitr, rmarkdown, testthat (>= 2.1.0)
VignetteBuilder  knitr
NeedsCompilation  no
Author  Arseniy Khvorov [aut, cre]
Maintainer  Arseniy Khvorov <khvorov45@gmail.com>
Repository  CRAN
Date/Publication  2019-11-26 21:40:02 UTC

R topics documented:

check_baseline ......................................................... 2
coef.sclr ............................................................... 3
find_prot_titre_val .................................................. 3
get_protection_level ................................................ 4
new_sclr ............................................................... 5
one_titre_data ....................................................... 6
predict.sclr ........................................................... 6
print.sclr .............................................................. 7
sclr ................................................................. 8
sclr_fit .............................................................. 10
sclr_ideal_data ...................................................... 11
Description

Fits the scaled logit model as well as logistic regression. Does a likelihood ratio test.

Usage

```r
check_baseline(
    formula = NULL,
    data = NULL,
    fit_sclr = NULL,
    fit_lr = NULL,
    conf_lvl = 0.95,
    verbose = TRUE
)
```

Arguments

- `formula`: Formula to use for model fitting.
- `data`: Optional dataframe.
- `fit_sclr`: Fit object returned by `sclr`.
- `fit_lr`: Fit object returned by `glm`.
- `conf_lvl`: Confidence level for the test
- `verbose`: Whether to print message based on test result.

Value

A `tibble` with a summary.

Examples

```r
library(sclr)
ll <- sclr_ideal_data(n = 50, theta = 1e6, seed = 20191104)
check_baseline(status ~ logHI, ll)
```
coef.sclr  

**ML estimate components**

**Description**

`coef` returns MLE’s. `vcov` returns the estimated variance-covariance matrix at MLE’s. `confint` returns the confidence interval. `model.matrix` returns the model matrix (x). `model.frame` returns the model frame (x and y in one matrix).

**Usage**

```r
## S3 method for class 'sclr'
coef(object, ...)

## S3 method for class 'sclr'
vcov(object, ...)

## S3 method for class 'sclr'
confint(object, parm, level = 0.95, ...)

## S3 method for class 'sclr'
model.matrix(object, ...)

## S3 method for class 'sclr'
model.frame(formula, ...)

## S3 method for class 'sclr'
logLik(object, ...)
```

**Arguments**

- `object`, `formula`
  - An object returned by `sclr`.
- `...`
  - Not used. Needed to match generic signature.
- `parm`
  - Parameter name, if missing, all parameters are considered.
- `level`
  - Confidence level.

**Description**

The search engine behind `get_protection_level`. Should not usually be necessary to call this directly.

**find_prot_titre_val**  

**Search function for scaled logit protection covariate levels**
Usage

find_prot_titre_val(
  fit,
  var_name,
  newdata = NULL,
  prot_var_name = "prot_point",
  lvl = 0.5,
  ci_level = 0.95,
  tol = 10^(-7)
)

Arguments

fit Object returned by sclr.

var_name Name of the covariate for which the protection values should be calculated. This name should appear in the formula of the call to sclr which was used to generate fit.

newdata A dataframe with all covariates except the one for which protection values should be calculated.

prot_var_name A variable name among those returned by predict.sclr which needs to equal lvl at the value of var_name that is supposed to be found.

lvl Protection level to find titre values for. Default is 0.5 (50%).

ci_level Confidence level for the calculated interval. Default is 0.95.

tol Tolerance. The values will be found numerically, once the algorithm converges within tol of lvl it stops looking. Default is 10^(−7).

Value

A dataframe. Will have the same variables as newdata with the addition of the var_name variable.

get_protection_level Protection level calculations

Description

Calculates covariate values corresponding to a particular protection level. Only accepts one covariate at a time, fixed values of all the others should be provided. The search engine is find_prot_titre_val.

Usage

get_protection_level(
  fit,
  var_name,
  newdata = NULL,
  lvl = 0.5,


```r
new_sclr

  ci_level = 0.95,
  tol = 10^(-7)

Arguments

  fit Object returned by `sclr`.
  var_name Name of the covariate for which to find values corresponding to a protection level. This name should appear in the formula in the call to `sclr` which was used to generate `fit`.
  newdata A dataframe with all covariates except the one for which protection values should be calculated. If there is only one covariate, can be left as `NULL` (the default).
  lvl Protection level to find covariate values for. Default is 0.5 (50%).
  ci_level Confidence level for the calculated interval. Default is 0.95.
  tol Tolerance. The values will be found numerically, once the algorithm converges within `tol` of `lvl` it stops looking. Default is 10^(-7).

Value

  A `tibble`. Will have the same variables as `newdata` with the addition of the `var_name` variable.

new_sclr

  Create a new `sclr` object

Description

  `new_sclr` creates the object `sclr` returns. `is_sclr` checks if the object is of class `sclr`.

Usage

  `new_sclr(fit, x, y, cl, mf, mt)`

  `is_sclr(fit)`

Arguments

  fit A list returned by `sclr_fit`.
  x Model matrix.
  y Model response.
  cl Call.
  mf Model frame.
  mt Model terms.

Value

  `sclr` object
```
### one_titre_data

**Simulated one-titre antibody data**

#### Description

A simulated dataset containing 5000 independent observations on antibody titres and the corresponding infection status. The data was simulated to resemble real influenza infection and haemagglutinin titre data.

#### Usage

```r
one_titre_data
```

#### Format

A data frame with 5000 observations and 2 variables:

- **logHI**: haemagglutinin-inhibiting (HI) titre. True simulated titre on a log scale.
- **status**: influenza infection status. 1 - infected. 0 - not infected

#### Model

The model behind the simulation was

\[ \lambda \ast (1 - f(\beta_0 + \beta_1 \ast HI)) \]

Where

- \( f \): Inverse logit function
- \( \lambda = 0.5 \)
- \( \beta_0 = -5 \)
- \( \beta_1 = 2 \)

### predict.sclr

**Predict method for scaled logit model x.**

#### Description

Returns only the protection estimates. The only supported interval is a confidence interval (i.e. the interval for the estimated expected value).

#### Usage

```r
## S3 method for class 'sclr'
predict(object, newdata, ci_lvl = 0.95, ...)
```
print.sclr

Arguments

object Object returned by sclr.
newdata A dataframe with all covariates. Names should be as they appear in the formula in the call to sclr.
ci_lvl Confidence level for the calculated interval.
... Not used. Needed to match generic signature.

Details

The model is

\[ P(Y = 1) = \lambda(1 - \logit^{-1}(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k)) \]

Where \( Y \) is the binary outcome indicator, (e.g. 1 - infected, 0 - not infected). \( X \) - covariate. \( k \) - number of covariates. This function calculates

\[ \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k \]

transformations at the covariate values found in newdata as well as the variance-covariance matrices of those transformations. This is used to calculate the confidence intervals at the given parameter values. The inverse logit transformation is then applied to point estimates and interval bounds.

Value

A tibble obtained by adding the following columns to newdata:

- prot_point_lin: Point estimate, low and high bounds of the linear transformation.
- prot_sd_lin: Estimated standard deviation of the linear transformation.
- prot_point: Inverse logit-transformed point estimate, low and high bounds of the linear transformation.

Description

Summarises a sclr object for printing. For a dataframe summary, see tidy.

Usage

```r
## S3 method for class 'sclr'
print(x, level = 0.95, ...)

## S3 method for class 'sclr'
summary(object, level = 0.95, ...)
```
**Arguments**

- **x, object** An object returned by `sclr`.
- **level** Confidence level for the intervals.
- **...** Not used. Needed to match generic signature.

---

**sclr** *Fits the scaled logit model*

---

**Description**

Used to fit the scaled logit model from Dunning (2006).

**Usage**

```r
sclr(
  formula,
  data = NULL,
  ci_lvl = 0.95,
  tol = 10^(-7),
  algorithm = c("newton-raphson", "gradient-ascent"),
  nr_iter = 2000,
  ga_iter = 2000,
  n_conv = 3,
  conventional_names = FALSE,
  seed = NULL
)
```

**Arguments**

- **formula** an object of class "formula": a symbolic description of the model to be fitted.
- **data** a data frame.
- **ci_lvl** Confidence interval level for the parameter estimates.
- **tol** Tolerance.
- **algorithm** Algorithms to run. "newton-raphson" or "gradient-ascent". If a character vector, the algorithms will be applied in the order they are present in the vector.
- **nr_iter** Maximum allowed iterations for Newton-Raphson.
- **ga_iter** Maximum allowed iterations for gradient ascent.
- **n_conv** Number of times the algorithm has to converge (to work around local maxima).
- **conventional_names** If TRUE, estimated parameter names will be (Baseline), (Intercept) and the column names in the model matrix. Otherwise - lambda, beta_0 and beta_ prefix in front of column names in the model matrix.
- **seed** Seed for the algorithms.
Details

The model is logistic regression with an added parameter for the top asymptote. That parameter is reported as \( \theta \) (or Baseline if \texttt{conventional_names = TRUE}). Note that it is reported on the logit scale. See \texttt{vignette("sclr-math")} for model specification, log-likelihood, scores and second derivatives. The main default optimisation algorithm is Newton-Raphson. Gradient ascent is used as a fallback by default. Computing engine behind the fitting is \texttt{sclr_fit}.

Value

An object of class \texttt{sclr}. This is a list with the following elements:

- \texttt{parameters} Maximum likelihood estimates of the parameter values.
- \texttt{covariance_mat} The variance-covariance matrix of the parameter estimates.
- \texttt{algorithm} Algorithm used.
- \texttt{algorithm_return} Everything the algorithm returned.
- \texttt{n_converge} The number of Newton-Raphson iterations (including resets) that were required for convergence.
- \texttt{x} Model matrix derived from \texttt{formula} and \texttt{data}.
- \texttt{y} Response matrix derived from \texttt{formula} and \texttt{data}.
- \texttt{call} The original call to \texttt{sclr}.
- \texttt{model} Model frame object derived from \texttt{formula} and \texttt{data}.
- \texttt{terms} Terms object derived from model frame.
- \texttt{ci} Confidence intervals of the parameter estimates.
- \texttt{log_likelihood} Value of log-likelihood calculated at the ML estimates of parameters.
- \texttt{formula} Passed formula.
- \texttt{data} Passed data.

Methods supported: \texttt{print}, \texttt{vcov}, \texttt{coef}, \texttt{model.frame}, \texttt{model.matrix}, \texttt{summary}, \texttt{predict}, \texttt{tidy} (\texttt{broom} package), \texttt{logLik}.

References


Examples

```r
library(sclr)
fit1 <- sclr(status ~ logHI, one_titre_data)
summary(fit1)
```
**sclr_fit**  
*Fitter function for the scaled logit model*

**Description**

Computing engine behind *sclr*.

**Usage**

```r
sclr_fit(  
y,  
x,  
tol = 10^(-7),  
algorithm = c("newton-raphson", "gradient-ascent"),  
nr_iter = 2000,  
ga_iter = 2000,  
n_conv = 3,  
conventional_names = FALSE,  
seed = NULL  
)
```

**Arguments**

- `y` A vector of observations.
- `x` A design matrix.
- `tol` Tolerance.
- `algorithm` Algorithms to run. "newton-raphson" or "gradient-ascent". If a character vector, the algorithms will be applied in the order they are present in the vector.
- `nr_iter` Maximum allowed iterations for Newton-Raphson.
- `ga_iter` Maximum allowed iterations for gradient ascent.
- `n_conv` Number of times the algorithm has to converge (to work around local maxima).
- `conventional_names` If TRUE, estimated parameter names will be (Baseline), (Intercept) and the column names in the model matrix. Otherwise - lambda, beta_0 and beta_ prefix in front of column names in the model matrix.
- `seed` Seed for the algorithms.

**Details**

The likelihood maximisation can use the Newton-Raphson or the gradient ascent algorithms.
sclr_ideal_data

Generate ideal data for the scaled logit model

Description

Allows variation of all parameters and the creation of an arbitrary number of covariates.

Usage

sclr_ideal_data(
  n = 1000,
  theta = 0,
  beta_0 = -5,
  covariate_list = list(logHI = list(gen_fun = function(n) rnorm(n, 2, 2), true_par = 2)),
  outcome_name = "status",
  seed = NULL,
  attach_true_vals = FALSE,
  attach_seed = FALSE
)

Arguments

n Number of observations.
theta Baseline risk parameter on the logit scale.
beta_0 Intercept of the linear part.
covariate_list A list in the form of name = list(gen_fun, true_par) where gen_fun is a function that takes n as an argument and returns a vector of observations, true_par is the true parameter value of that covariate. See examples.
outcome_name Name to give to the outcome
seed Seed to set. If NULL, no seed will be set.
attach_true_vals, attach_seed Whether to attach additional attributes.

Value

A tibble.

Examples

# One titre
one_titre <- sclr_ideal_data(
  covariate_list = list(
    logHI = list(gen_fun = function(n) rnorm(n, 2, 2), true_par = 2)
  )
)
tidy.sclr

```r
sclr(status ~ logHI, one_titre) # Verify

# Two titres
two_titre <- sclr_ideal_data(
  covariate_list = list(
    logHI = list(gen_fun = function(n) rnorm(n, 2, 2), true_par = 2),
    logNI = list(gen_fun = function(n) rnorm(n, 2, 2), true_par = 1)
  )
)
sclr(status ~ logHI + logNI, two_titre) # Verify
```

sclr_log_likelihood  

**Log-likelihood**

**Description**

Computes the log-likelihood of the scaled logit model at a given set of parameter estimates (or the MLE if `pars` is not supplied). Either `fit` or `x`, `y` and `pars` need to be supplied.

**Usage**

```r
sclr_log_likelihood(fit = NULL, x = NULL, y = NULL, pars = NULL)
```

**Arguments**

- `fit`: An object returned by `sclr`. Or a list with parameters, `x` and `y` entries corresponding to the parameter matrix, model matrix and model response.
- `x`: Model matrix. Will be taken from `fit` if `fit` is provided.
- `y`: Model response. Will be taken from `fit` if `fit` is provided.
- `pars`: A named vector of parameter values. Will be taken from `fit` if `fit` is provided.

tidy.sclr  

**Tidy a sclr object.**

**Description**

Summarises the objects returned by `sclr` into a tibble.

**Usage**

```r
## S3 method for class 'sclr'
tidy(x, ci_level = 0.95, ...)
```
two_titre_data

Arguments

x  An object returned by sclr.

\texttt{ci\_level}  Confidence level for the intervals.

... Not used. Needed to match generic signature.

Value

A \texttt{tibble} with one row per model parameter. Columns:

- \texttt{term}  Name of model parameter.
- \texttt{estimate}  Point estimate.
- \texttt{std\_error}  Standard error.
- \texttt{conf\_low}  Lower bound of the confidence interval.
- \texttt{conf\_high}  Upper bound of the confidence interval.

\begin{verbatim}
two_titre_data  Simulated two-titre antibody data
\end{verbatim}

Description

A simulated dataset containing 5000 independent observations on antibody titres and the corresponding infection status. The data was simulated to resemble real influenza infection and haemagglutinin + neuraminidase titre data.

Usage

two_titre_data

Format

A data frame with 5000 observations and 3 variables:

- \texttt{logHI}  haemagglutinin-inhibiting (HI) titre. True simulated titre on a log scale.
- \texttt{logNI}  neuraminidase-inhibiting titre. True simulated titre on a log scale.
- \texttt{status}  influenza infection status. 1 - infected. 0 - not infected

Model

The model behind the simulation was

\[
\lambda * (1 - f(\beta_0 + \beta_1 * HI + \beta_2 * NI))
\]

Where

- \(f\) - Inverse logit function
• $\lambda = 0.5$
• $\beta_0 = -7.5$
• $\beta_1 = 2$
• $\beta_2 = 2$
# Index

*Topic **datasets**

- one_titre_data, 6
- two_titre_data, 13

broom, 9

- check_baseline, 2
- coef, 9
- coef.sclr, 3
- confint.sclr (coef.sclr), 3

find_prot_titre_val, 3, 4

glm, 2

is_sclr (new_sclr), 5

logLik, 9
- logLik.sclr (coef.sclr), 3

model.frame, 9
- model.frame.sclr (coef.sclr), 3

model.matrix, 9
- model.matrix.sclr (coef.sclr), 3

new_sclr, 5

one_titre_data, 6

predict, 9
- predict.sclr, 4, 6

print, 9
- print.sclr, 7

sclr, 2–5, 7, 8, 10, 12, 13
- sclr_fit, 5, 9, 10
- sclr_ideal_data, 11
- sclr_log_likelihood, 12

summary, 9
- summary.sclr (print.sclr), 7

- tibble, 2, 5, 7, 11–13
- tidy, 7, 9
- tidy.sclr, 12
- two_titre_data, 13

vcov, 9
- vcov.sclr (coef.s clr), 3

15