Package ‘selfmade’

April 15, 2021

Title  Selective Inference for Mixed and Additive Model Estimators

Version  0.1


Depends  R (>= 3.5)

License  GPL (>= 2)

Encoding  UTF-8

LazyData  false

RoxygenNote  7.1.1

Imports  Matrix, lme4, mgcv

Suggests  knitr, rmarkdown, gamm4, lmerTest

VignetteBuilder  knitr

NeedsCompilation  no

Author  David Ruegamer [aut, cre]

Maintainer  David Ruegamer <david.ruegamer@gmail.com>

Repository  CRAN

Date/Publication  2021-04-15 08:30:02 UTC

R topics documented:

mocasin ................................................................. 2
print.selfmade ....................................................... 5

Index  6
mocasin function which computes selective p-values and intervals for `gamm4` and `merMod` objects

Description

Function which computes selective p-values and intervals for `gamm4` and `merMod` objects

Usage

```r
mocasin(
  mod,
  checkFun,
  this_y = NULL,
  nrSamples = 1000,
  bayesian = TRUE,
  varInTestvec = c("est", "minMod", "varY", "supplied"),
  varForSampling = c("est", "minMod", "varY", "supplied"),
  VCOV_vT = NULL,
  VCOV_sampling = NULL,
  conditional = TRUE,
  name = NULL,
  nrlocs = 7,
  complete_effect = NULL,
  which = NULL,
  vT = NULL,
  G = NULL,
  efficient = TRUE,
  trace = TRUE
)
```

Arguments

- **mod**
  - an object of class `merMod` or result of `gamm4` function

- **checkFun**
  - a function of `y`, a vector of the same length as the original response vector which returns TRUE or FALSE depending on whether the selection event for a given `y` corresponds to the original model selection. See the example for more details.

- **this_y**
  - original response vector (explicit reference may be necessary for certain model classes)

- **nrSamples**
  - integer; the number of Monte Carlo samples to be used for inference (defaults to 1000)

- **bayesian**
  - logical; whether or not to use a bayesian type covariance

- **varInTestvec**
  - for expert use only; variance used in the test vector definition
varForSampling  variance used for inference; per default the estimated variance of mod is used. Other options are a conservative estimate based on the variance of the response is used (“varY”) or to supply a numeric value to base inference on a customize variance.

VCOV_vT  for expert use only; VCOV used in the test vector definition.

VCOV_sampling  covariance matrix of dimension of the response used for inference; per default the estimated covariance of mod is used. Otherwise a matrix must be supplied on which basis inference is conducted. If the true covariance is unknown, an conservative alternative to plugging in the estimator is given by using the co-

variance of the refitted mixed model, for which all fixed effects but the intercept are excluded.

conditional  logical; determines whether to use the conditional or marginal approach when mod is of class merMod, i.e., inference is sought for a linear mixed model.

name  character; for the gamm4-case: the name of the covariate, for which inference is done.

nrlocs  integer; for the gamm4-case: the number of locations tested for non-linear effects.

complete_effect  list of logical values for each name; TRUE performs a (conservative) test whether the whole spline has a significant influence after accounting for all other effects.

which  integer; for the merMod-case: defining the effect for which inference is done.

vT  list of vectors (optional); if inference is sought for a customized test vector, this argument can be used.

G  true random effect covariance (optional).

efficient  logical; whether or not to compute the test statistic based on an (efficient) weighted LS estimator instead of a OLS estimator for the marginal model.

trace  logical; if TRUE, a progress bar is printed in the console.

Details

Note that the additive and conditional mixed model approach currently only works for a diagonal error covariance.

Value

An object of class selfmade with corresponding print method.

Examples

library(lme4)
if(require(lmerTest)){

##### BASED ON lmerTest HELP PAGE ############
# define function to fit a model based on response
modFun <- function(y)
{
  ham$y <- y
}
lmer(y ~ Gender + Information * Product + (1 | Consumer) +
     (1 | Product), data=ham)

# define a function to select a model (based on a model)
selfFun <- function(mod) step(mod)

# define a function which extracts the results of the selection procedure
extractSelFun <- function(this_mod){
  this_mod <- attr(this_mod, "model")
  if(class(this_mod)="lm")
    return(attr(this_mod$coefficients, "names"))
  else
    return(c(names(fixef(this_mod)), names(getME(this_mod, "theta"))))
}

## backward elimination of non-significant effects:
(step_result <- selFun(modFun(ham$Informed.liking)))
attr(step_result, "model")

## Elimination tables for random- and fixed-effect terms:
(sel <- extractSelFun(step_result))

## Now we can finally define the function checking the congruency
## with the original selection

checkFun <- function(yb){
  this_mod <- modFun(yb)
  setequal( extractSelFun(selFun(this_mod)), sel )
}

# Now let's compute valid p-values conditional on the selection
# Not run:
res <- mocasin(attr(step_result, "model"), this_y = ham$Informed.liking,
               checkFun = checkFun, which = 1, nrSamples = 8, trace = FALSE)

# print(res)

# End(Not run)

# gamm4 example similar to the one from gamm4 help page
if(require(gamm4)){
  set.seed(0)
  dat <- gamSim(1, n = 500, scale = 2) ## simulate 4 term additive truth
  dat$y <- 3 + dat$x0^2 + rnorm(n = 500)
  br <- gamm4(y~ s(x0) + s(x1), data = dat)
  summary(br$gam) ## summary of gam
# do not use any selection
checkFun <- function(yb) TRUE

## Not run:
res <- mocasin(br, this_y = dat$y,
    checkFun = checkFun,
    nrlocs = c(0.7),
    nrSamples = 100)

# print result
res

## End(Not run)
}

---

**print.selfmade**

**Generic methods for selfmade objects**

**Description**

Generic methods which can be used for objects fitted with the `mocasin` function.

**Usage**

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

**Arguments**

- `x` selfmade object
- `...` further arguments, currently unused.

**Value**

prints the object of class `selfmade` to console.
Index

mocasin, 2

print.selfmade, 5