Package ‘MATA’

February 15, 2019

Title Model-Averaged Tail Area Wald (MATA-Wald) Confidence Interval

Version 0.4

Description Calculates Model-Averaged Tail Area Wald (MATA-Wald) confidence intervals, which are constructed using single-model estimators and model weights. See Turek and Fletcher (2012) <doi:10.1016/j.csda.2012.03.002> for details.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation no

Author Daniel Turek [aut, cre]

Maintainer Daniel Turek <danielturek@gmail.com>

Repository CRAN

Date/Publication 2019-02-15 16:10:03 UTC

R topics documented:

`mata.wald` *Model-Averaged Tail Area Wald (MATA-Wald) Confidence Interval*

Description

A function for computing the Model-Averaged Tail Area Wald (MATA-Wald) confidence interval, constructed using single-model estimators and model weights.

Usage

```
mata.wald(theta.hats, se.theta.hats, model.weights, mata.t, residual.dfs,
  alpha = 0.025, normal.lm)
```

Arguments

- `theta.hats` — A numeric vector containing the parameter estimates under each candidate model.
- `se.theta.hats` — A numeric vector containing the estimated standard error of each value in `theta.hats`.
- `model.weights` — A vector containing the model weights for each candidate model. Calculated from an information criterion, such as AIC or BIC. All model weights must be non-negative, and sum to one.
- `mata.t` — Logical. TRUE for the normal linear model case, and FALSE otherwise. When TRUE, the argument `residual.dfs` must also be supplied.
- `residual.dfs` — A vector containing the residual (error) degrees of freedom under each candidate model. This argument must be provided when `mata.t = TRUE`.
- `alpha` — The desired lower and upper error rate. The value 0.025 corresponds to a 95% MATA-Wald confidence interval, and 0.05 to a 90% interval. Must be between 0 and 0.5. Default value is 0.025.
- `normal.lm` — Provided only for backward-compatibility. This argument has been deprecated, and replaced by `mata.t`.

Details

`mata.wald` may be used to construct model-averaged confidence intervals, using the Model-Averaged Tail Area (MATA) construction (see Turek and Fletcher (2012) for details). The idea underlying this construction is similar to that of a model-averaged Bayesian credible interval. This function returns the lower and upper confidence limits of a MATA-Wald interval.

Two usages are supported. For the normal linear model, or any other model where a t-based interval is appropriate (e.g., quasi-poisson), using option `mata.t = TRUE` generates a MATA-Wald confidence interval corresponding to the solutions of equations (2) and (3) of Turek and Fletcher (2012). The argument `residual.dfs` is required for this usage.

When the sampling distribution for the estimator is asymptotically normal (e.g. MLEs), possibly after a transformation, use option `mata.t = FALSE`. This generates a MATA-Wald confidence interval, possibly on a transformed scale, where back-transformation of both confidence limits may be necessary. This corresponds to solutions to the equations in Section 3.2 of Turek and Fletcher (2012).

Author(s)

Daniel Turek

References

Turek, D. and Fletcher, D. (2012). Model-Averaged Wald Confidence Intervals. Computational Statistics and Data Analysis, 56(9), p.2809-2815.

Fletcher, D. (2018). Model Averaging. Berlin, Heidelberg: Springer Briefs in Statistics.

Examples

```
# Normal linear prediction:
# Generate single-model Wald and model-averaged MATA-Wald 95% confidence intervals
#
# Data 'y', covariates 'x1' and 'x2', all vectors of length 'n'.
# 'y' taken to have a normal distribution.
# 'x1' specifies treatment/group (factor).
# 'x2' a continuous covariate.
#
# Take the quantity of interest (theta) as the predicted response
# (expectation of y) when x1=1 (second group/treatment), and x2=15.

n = 20                              # 'n' is assumed to be even
x1 = c(rep(0,n/2), rep(1,n/2))      # two groups: x1=0, and x1=1
x2 = rnorm(n, mean=10, sd=3)
y = rnorm(n, mean = 3*x1 + 0.1*x2)  # data generation

x1 = factor(x1)
m1 = glm(y ~ x1)                    # using 'glm' provides AIC values.
m2 = glm(y ~ x1 + x2)               # using 'lm' doesn't.
aic = c(m1$aic, m2$aic)
delta.aic = aic - min(aic)
model.weights = exp(-0.5*delta.aic) / sum(exp(-0.5*delta.aic))
residual.dfs = c(m1$df.residual, m2$df.residual)

p1 = predict(m1, se=TRUE, newdata=list(x1=factor(1), x2=15))
p2 = predict(m2, se=TRUE, newdata=list(x1=factor(1), x2=15))
theta.hats = c(p1$fit, p2$fit)
se.theta.hats = c(p1$se.fit, p2$se.fit)

#  AIC model weights
model.weights

#  95% Wald confidence interval for theta (under Model 1)
theta.hats[1] + c(-1,1)*qt(0.975, residual.dfs[1])*se.theta.hats[1]

#  95% Wald confidence interval for theta (under Model 2)
theta.hats[2] + c(-1,1)*qt(0.975, residual.dfs[2])*se.theta.hats[2]

#  95% MATA-Wald confidence interval for theta (model-averaging)
mata.wald(theta.hats=theta.hats, se.theta.hats=se.theta.hats,
        model.weights=model.weights, mata.t=TRUE, residual.dfs=residual.dfs)
```

Index