Package ‘stabreg’

June 6, 2019

Type Package

Title Linear Regression with the Stable Distribution

Version 0.1.2

Maintainer Oleg Kopylov <okopy@protonmail.com>

Description Efficient regression for heavy-tailed and skewed data following a stable distribution. Generalized regression where the skewness and tail parameter of residuals are dependent on regressors is also available. Includes fast calculation of stable densities. Calculation of densities is based on efficient numerical methods from Ament and O'Neil (2017) <doi:10.1007/s11222-017-9725-y>. Parts of the code have been ported to C from Ament's 'Matlab' code available at <https://gitlab.com/s_ament/qastable>.

Encoding UTF-8

LazyData true

License GPL-3

NeedsCompilation yes

Imports numDeriv

Author Oleg Kopylov [aut, cre],
Sebastian Ament [ctb]

Repository CRAN

Date/Publication 2019-06-06 14:20:03 UTC

R topics documented:

AIC.stabreg 2
predict.stabreg 2
print.stabreg 3
stable_glm 3
stable_lm 4
stable_mle_fit 6
stable_pdf 6

Index **7**

AIC.stabreg — *AIC.stabreg*

Description

Akaike's Information Criterion

Usage

```
## S3 method for class 'stabreg'
AIC(object, ..., k = 2)
```

Arguments

`object`	a stabreg object
`...`	not used
`k`	AIC penalty

predict.stabreg — *predict.stabreg*

Description

Calculate predictions

Usage

```
## S3 method for class 'stabreg'
predict(object, ...)
```

Arguments

`object`	a stabreg object
`...`	passed to model.matrix

`print.stabreg` *print.stabreg*

Description

print method

Usage

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

Arguments

`x`	a stabreg object
`...`	not used

`stable_glm` *Generalized linear-model fitting with Stable residuals*

Description

Fitting of a generalized linear model with stable residuals. Allows for regression formulas for all 4 parameters of the stable distribution. NAs not allowed.

Usage

```
stable_glm(y_name, formulas, data, output_se, calc_confbounds, conf, trace, optim_control)
```

Arguments

`y_name`	character string denoting column name in input data, containing dependent variable
`formulas`	list for formulas for the regression. See Details
`data`	data.frame
`output_se`	logical - whether to calculate standard errors (will entail calculation and inversion of final Hessian)
`calc_confbounds`	logical - whether to calculate exact confidence bounds. See Details
`conf`	confidence level for confidence bounds. Default 0.95
`trace`	trace level
`optim_control`	list passed to `nlminb`'s control argument

Details

Lowest possible estimate for alpha is 1.1.

The formulas argument is a list of formulas, where each name in the list is either "loc", "scale", "beta", or "alpha". For each of these, a formula is supplied without any dependent variable. E.g. in a regression for the alpha parameter the formulas list will look like `formulas = list( alpha = formula(~ x1 + x2) )`

By default, `output_se = TRUE`, which will calculate standard errors and approximate confidence bounds. These will be approximate in the sense that the likelihood will typically not be symmetric, and thus these confidence bounds will only provide an indicative measure of confidence. To calculate exact confidence bounds use `calc_confbounds = TRUE`. However, exact confidence bounds calculation is very time consuming as this requires repeated computation of profile likelihoods for each regression variable. If N is the number of regressors in the model, then the expected time to compute all confidence bounds is about 200N times the time it takes to just fit the model. Currently, confidence bounds are calculated only for the regressors and not the parameters of the distribution of the residuals (i.e. alpha, beta, and scale).

Note that standard errors and t-scores for alpha, beta, and scale parameters are calculated in transformed space and cannot be interpreted as-is. All confidence bounds (both approximate and exact), however, are transformed back to the original space and can be interpreted as-is.

In the case that alpha is estimated to be numerically equal to 2, beta is automatically set to 0. In this boundary case calculation of standard errors will fail and return the default maximum range for alpha, i.e. 1.1 to 2. This should be interpreted as the residuals having a Normal distribution, in which case OLS regression would be preferable.

Examples

```
# generate some data: y = 4x - 1 + epsilon
# where epsilon is heavy-tailed student-t with 5 df
set.seed(123)
df <- data.frame(x = rnorm(500))
df$y <- 4 * df$x - 1 + rt(500, df = 5)

# regress both location and skew:
formulas <- list(
    "loc" = formula( ~ x),
    "beta" = formula( ~ x)
)

sfit <- stable_glm("y", formulas, data = df)
print(sfit)
```

---

stable_lm *Linear-model fitting with Stable residuals*

Description

Fitting of a linear model with stable residuals. Regresses only location of the distribution. NAs not allowed.

Usage

```
stable_lm(formula, data, trace, output_se, calc_confbounds, conf, optim_control)
```

Arguments

formula	regression formula for location
data	data.frame
trace	trace level
output_se	logical - whether to calculate standard errors (will entail calculation and inversion of final Hessian)
calc_confbounds	logical - whether to calculate exact confidence bounds. See Details
conf	confidence level for confidence bounds. Default 0.95
optim_control	list passed to nlminb's control argument

Details

Lowest possible estimate for alpha is 1.1.

By default, `output_se = TRUE`, which will calculate standard errors and approximate confidence bounds. These will be approximate in the sense that the likelihood will typically not be symmetric, and thus these confidence bounds will only provide an indicative measure of confidence. To calculate exact confidence bounds use `calc_confbounds = TRUE`. However, exact confidence bounds calculation is very time consuming as this requires repeated computation of profile likelihoods for each regression variable. If N is the number of regressors in the model, then the expected time to compute all confidence bounds is about 200N times the time it takes to just fit the model. Currently, confidence bounds are calculated only for the regressors and not the parameters of the distribution of the residuals (i.e. alpha, beta, and scale).

Note that standard errors and t-scores for alpha, beta, and scale parameters are calculated in transformed space and cannot be interpreted as-is. All confidence bounds (both approximate and exact), however, are transformed back to the original space and can be interpreted as-is.

In the case that alpha is estimated to be numerically equal to 2, beta is automatically set to 0. In this boundary case calculation of standard errors will fail and return the default maximum range for alpha, i.e. 1.1 to 2. This should be interpreted as the residuals having a Normal distribution, in which case OLS regression would be preferable.

Examples

```
# generate some data: y = 4x - 1 + epsilon
# where epsilon is heavy-tailed student-t with 5 df
set.seed(123)
df <- data.frame(x = rnorm(1000))
df$y <- 4 * df$x - 1 + rt(1000, df = 5)

sfit <- stable_lm(y ~ x, data = df)
print(sfit)
```

`stable_mle_fit` *Fit a stable distribution to a sample using maximum likelihood*

Description

Fit a stable distribution to a sample using maximum likelihood

Usage

```
stable_mle_fit(x, init_vals, trace)
```

Arguments

`x`	sample vector
`init_vals`	initial guess for parameters. Defaults to NULL in which case these are set to defaults
`trace`	trace level

`stable_pdf` *Compute values of a (normalized) stable density*

Description

Compute stable pdf values. Admissible range for beta is -1 to 1. If `b` is not 0, admissible values of alpha are `0.5 < a < 0.9` and `1.1 < a <= 2`.

Usage

```
stable_pdf(x, a, b)
```

Arguments

`x`	values at which to evaluate the density
`a`	alpha value. Either a scalar or vector of length equal to `x` if different values for each observations are assumed
`b`	beta value. Either a scalar or vector of length equal to `x` if different values for each observations are assumed

Index

AIC.stabreg, 2

predict.stabreg, 2
print.stabreg, 3

stable_glm, 3
stable_lm, 4
stable_mle_fit, 6
stable_pdf, 6