Package ‘esreg’

May 13, 2023

Type Package

Title Joint Quantile and Expected Shortfall Regression

Version 0.6.2

Date 2023-05-12

Description

Simultaneous modeling of the quantile and the expected shortfall of a response variable given a set of covariates, see Dimitriadis and Bayer (2019) <doi:10.1214/19-EJS1560>.

License GPL-3

Encoding UTF-8

Imports quantreg, Rcpp, stats, Formula

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.2.3

NeedsCompilation yes

Author Sebastian Bayer [aut, cre],
Timo Dimitriadis [aut]

Maintainer Sebastian Bayer <sebastian.bayer@uni-konstanz.de>

Repository CRAN

Date/Publication 2023-05-13 08:30:02 UTC

R topics documented:

esreg *Joint Quantile and Expected Shortfall Regression*

Description

Estimates a joint linear regression model for the pair (VaR, ES):

$$Q_\alpha(Y|Xq) = Xq'\beta_q$$

$$ES_\alpha(Y|Xe) = Xe'\beta_e$$

Usage

```
esreg(...)

## S3 method for class 'formula'
esreg(
  formula,
  data = parent.frame(),
  alpha,
  g1 = 2L,
  g2 = 1L,
  early_stopping = 10,
  ...
)

## Default S3 method:
esreg(xq, xe, y, alpha, g1 = 2L, g2 = 1L, early_stopping = 10, ...)
```

Arguments

`...`	Further arguments (does not apply here)
`formula`	Formula: y ~ x1 + x2 ... | x1 + x2 ... where the first part after the response variable specifies the quantile equation and the second the expected shortfall part. If only one set of regressors is provided it is used for both model specifications.
`data`	data.frame that holds the variables
`alpha`	Probability level
`g1`	1, 2 (see G1_fun, G1_prime_fun), defaults to 1
`g2`	1, 2, 3, 4, 5 (see G2_curly_fun, G2_fun, G2_prime_fun). defaults to 2
`early_stopping`	Stop the iterated local search if there is no improvement in early_stopping steps.
`xq`	Explanatory variables for the quantile regression equation
`xe`	Explanatory variables for the expected shortfall regression equation
`y`	Response vector

Value

An esreg object

References

A Joint Quantile and Expected Shortfall Regression Framework

See Also

`vcov.esreg` for covariance estimation

Examples

```
# Simulate data (DGP-(2) in the linked paper)
set.seed(0)
x <- rchisq(1000, df=1)
y <- -x + (1 + 0.5 * x) * rnorm(1000)

# True quantile and expected shortfall regression parameters (for alpha=0.025)
alpha=0.025
true_pars <- c(-1.959964, -1.979982, -2.337803, -2.168901)

# Estimate the model using the standard settings
fit <- esreg(y ~ x, alpha=alpha)

# Compare the different variance-covariance estimators
cov1 <- vcov(object=fit, sparsity="iid", sigma_est="ind")
cov2 <- vcov(object=fit, sparsity="nid", sigma_est="scl_N")
cov3 <- vcov(object=fit, sparsity="nid", sigma_est="scl_sp")

print("Comparison of the variance-covariance estimators")
print(cbind(Truth=true_pars,
            Estimate=coef(fit),
            SE_iid_ind=sqrt(diag(cov1)),
            SE_nid_N=sqrt(diag(cov2)),
            SE_nid_sp=sqrt(diag(cov3))))

# Compares estimates using different G2 functions
fit1 <- esreg(y ~ x, alpha=alpha, g2=1)
fit2 <- esreg(y ~ x, alpha=alpha, g2=2)
fit3 <- esreg(y ~ x, alpha=alpha, g2=3)
fit4 <- esreg(y ~ x, alpha=alpha, g2=4)
fit5 <- esreg(y ~ x, alpha=alpha, g2=5)
fits <- sapply(list(fit1, fit2, fit3, fit4, fit5), coef)
colnames(fits) <- sapply(1:5, function(i) esreg:::.G_function_names(1, i)[2])
print("Comparison of the five G2 functions")
print(rbind(Truth=true_pars, t(fits)))

# Usage of different covariates
x <- rchisq(1000, df=1)
noise <- rnorm(1000)
y <- -x + (1 + 0.5 * x) * rnorm(1000)
```

```
fit <- esreg(y ~ x | x + noise, alpha=0.025)
print("Using different covariates for VaR and ES")
print(summary(fit))
```

esr_loss *Joint Loss Function*

Description

Computes the joint (VaR, ES) loss

Usage

```
esr_loss(r, q, e, alpha, g1 = 2L, g2 = 1L, return_mean = TRUE)
```

Arguments

r	Vector of returns
q	Vector of quantiles
e	Vector of expected shortfalls
alpha	Probability level
g1	1, 2, see G1_fun
g2	1, 2, 3, 4, 5, see G2_curly_fun, G2_fun
return_mean	If TRUE returns the average tick loss, else the individual values

References

Fissler and Ziegel (2016)

estfun.esreg *Estimating function*

Description

This function matches the estfun function of the sandwich package and returns the estimating functions for the fitted model. It can for instance be used for an OPG estimator of the sigma matrix. For esreg, the dimension of the estimating functions is n x (kq + ke).

Usage

```
estfun.esreg(x, ...)
```

Arguments

x	An esreg object
...	Further arguments (does not apply here)

lambda_matrix *Lambda Matrix*

Description

Estimate the lambda matrix.

Usage

```
lambda_matrix(object, sparsity, bandwidth_estimator, misspec)
```

Arguments

`object` An esreg object

`sparsity` The estimator to be used for the sparsity in Λ, see density_quantile_function

- iid - Piecewise linear interpolation of the distribution
- nid - Hendricks and Koenker sandwich

`bandwidth_estimator` The bandwidth estimator to be used for the iid and nid sparsity estimator, see density_quantile_function

- Bofinger
- Chamberlain
- Hall-Sheather

`misspec` if TRUE, the estimator accounts for potential misspecification in the model

sigma_matrix *Sigma Matrix*

Description

Estimate the sigma matrix.

Usage

```
sigma_matrix(object, sigma_est, misspec)
```

Arguments

`object` An esreg object

`sigma_est` The estimator to be used for Σ, see conditional_truncated_variance

- ind - Variance over all negative residuals
- scl_N - Scaling with the normal distribution
- scl_sp - Scaling with the kernel density function

`misspec` if TRUE, the estimator accounts for potential misspecification in the model

vcov.esreg *Covariance Estimation*

Description

Estimate the variance-covariance matrix of the joint (VaR, ES) estimator

Usage

```
## S3 method for class 'esreg'
vcov(object, method = "asymptotic", ...)
```

Arguments

`object`	An esreg object
`method`	For asymptotic use vcovA, for boot use vcovB
`...`	All possible values which can be passed to vcovA and vcovB

vcovA *Asymptotic Covariance Estimation*

Description

Estimate the variance-covariance matrix of the joint (VaR, ES) estimator by the sandwich formula:

$$\lambda^{-1}\Sigma\lambda^{-1}$$

Several estimators are available for both matrices and the default options are selected to take into account possible misspecifications in the underlying data.

Usage

```
vcovA(
  object,
  sigma_est = "scl_sp",
  sparsity = "nid",
  misspec = TRUE,
  bandwidth_estimator = "Hall-Sheather"
)
```

Arguments

`object` An esreg object

`sigma_est` The estimator to be used for Σ, see conditional_truncated_variance

- ind - Variance over all negative residuals
- scl_N - Scaling with the normal distribution
- scl_sp - Scaling with the kernel density function

`sparsity` The estimator to be used for the sparsity in Λ, see density_quantile_function

- iid - Piecewise linear interpolation of the distribution
- nid - Hendricks and Koenker sandwich

`misspec` if TRUE, the estimator accounts for potential misspecification in the model

`bandwidth_estimator`
The bandwidth estimator to be used for the iid and nid sparsity estimator, see density_quantile_function

- Bofinger
- Chamberlain
- Hall-Sheather

`vcovB` *Bootstrap Covariance Estimation*

Description

Estimate the variance-covariance matrix of the joint (VaR, ES) estimator using the bootstrap.

Usage

```
vcovB(object, bootstrap_method = "iid", B = 1000)
```

Arguments

`object` An esreg object

`bootstrap_method`
The bootstrap sampling scheme to be used

- iid - The iid bootstrap of Efron (1979)

`B` The number of bootstrap iterations

Index