Package 'ACV'

April 5, 2022

Title Optimal Out-of-Sample Forecast Evaluation and Testing under Stationarity

Version 1.0.2

Description Package 'ACV' (short for Affine Cross-Validation) offers an improved time-series cross-validation loss estimator which utilizes both in-sample and out-of-sample forecasting performance via a carefully constructed affine weighting scheme. Under the assumption of stationarity, the estimator is the best linear unbiased estimator of the out-of-sample loss. Besides that, the package also offers improved versions of Diebold-Mariano and Ibragimov-Muller tests of equal predictive ability which deliver more power relative to their conventional counterparts. For more information, see the accompanying article Stanek (2021) <doi:10.2139/ssrn.3996166>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.1.2

Imports forecast, Matrix, methods, stats

Depends

Suggests testthat

NeedsCompilation no

Author Filip Stanek [aut, cre]

Maintainer Filip Stanek <stanek.fi@gmail.com>

Repository CRAN

Date/Publication 2022-04-05 09:40:13 UTC

R topics documented:

estimateL 2
testL 3
tsACV 5

Index **7**

`estimateL` *Estimate out-of-sample loss*

Description

Function `estimateL()` estimates the out-of-sample loss of a given algorithm on specified time-series. By default, it uses the optimal weighting scheme which exploits also the in-sample performance in order to deliver a more precise estimate than the conventional estimator.

Usage

```
estimateL(
  y,
  algorithm,
  m,
  h = 1,
  v = 1,
  xreg = NULL,
  lossFunction = function(y, yhat) {     (y - yhat)^2 },
  method = "optimal",
  Phi = NULL,
  bw = NULL,
  rhoLimit = 0.99,
  ...
)
```

Arguments

`y`	Univariate time-series object.
`algorithm`	Algorithm which is to be applied to the time-series. The object which the algorithm produces should respond to `fitted` and `forecast` methods. Alternatively in the case of more complex custom algorithms, the algorithm may be a function which takes named arguments ("yInSample", "yOutSample", "h") or ("yInSample", "yOutSample", "h", "xregInSample", "xregOutSample") as inputs and produces a list with named elements ("yhatInSample", "yhatOutSample") containing vectors of in-sample and out-of-sample forecasts.
`m`	Length of the window on which the algorithm should be trained.
`h`	Number of predictions made after a single training of the algorithm.
`v`	Number of periods by which the estimation window progresses forward once the predictions are generated.
`xreg`	Matrix of exogenous regressors supplied to the algorithm (if applicable).
`lossFunction`	Loss function used to compute contrasts (defaults to squared error).
`method`	Can be set to either `"optimal"` for the estimator which optimally utilizes also the in-sample performance or `"convetional"` for the conventional loss estimator.

Phi	User can also directly supply Phi; the matrix of contrasts produced by tsACV. In this case parameters: y, algorithm, m, h, v, xreg, lossFunction are ignored.
bw	Bandwidth for the long run variance estimator. If NULL, bw is selected according to (3/4)*n^(1/3).
rhoLimit	Parameter rhoLimit limits to the absolute value of the estimated rho coefficient. This is useful as estimated values very close to 1 might cause instability.
...	Other parameters passed to the algorithm.

Value

List containing loss estimate and its estimated variance along with some other auxiliary information like the matrix of contrasts Phi and the weights used for computation.

Examples

```
set.seed(1)
y <- rnorm(40)
m <- 36
h <- 1
v <- 1
estimateL(y, forecast::Arima, m = m, h = h, v = v)
```

---

testL *Test equality of out-of-sample losses of two algorithms*

Description

Function testL() tests the null hypothesis of equal predictive ability of algorithm1 and algorithm2 on time series y. By default, it uses the optimal weighting scheme which exploits also the in-sample performance in order to deliver more power than the conventional tests.

Usage

```
testL(
  y,
  algorithm1,
  algorithm2,
  m,
  h = 1,
  v = 1,
  xreg = NULL,
  lossFunction = function(y, yhat) {     (y - yhat)^2 },
  method = "optimal",
  test = "Diebold-Mariano",
  Ha = "!=0",
  Phi = NULL,
```

```
  bw = NULL,
  groups = 2,
  rhoLimit = 0.99,
  ...
)
```

Arguments

y	Univariate time-series object.
algorithm1	First algorithm which is to be applied to the time-series. The object which the algorithm produces should respond to `fitted` and `forecast` methods. Alternatively in the case of more complex custom algorithms, the algorithm may be a function which takes named arguments ("yInSample", "yOutSample", "h") or ("yInSample", "yOutSample", "h", "xregInSample", "xregOutSample") as inputs and produces a list with named elements ("yhatInSample", "yhatOutSample") containing vectors of in-sample and out-of-sample forecasts.
algorithm2	Second algorithm. See above.
m	Length of the window on which the algorithm should be trained.
h	Number of predictions made after a single training of the algorithm.
v	Number of periods by which the estimation window progresses forward once the predictions are generated.
xreg	Matrix of exogenous regressors supplied to the algorithm (if applicable).
lossFunction	Loss function used to compute contrasts (defaults to squared error).
method	Can be set to either `"optimal"` for the test which optimally utilizes also the in-sample performance or `"convetional"` for the conventional test.
test	Type of the test which is to be executed. Can attain values `"Diebold-Mariano"` for the canonical test of equal predictive ability or `"Ibragimov-Muller"` for the sub-sampling t-test.
Ha	Alternative hypothesis. Can attain values `"!=0"` for two sided test or `"<0"` and `">0"` for one sided tests.
Phi	User can also directly supply `Phi=Phi1-Phi2`; the matrix of contrasts differentials produced by `tsACV`. In this case parameters: `y`, `algorithm`, `m`, `h`, `v`, `xreg`, `lossFunction` are ignored.
bw	Applicable to `"Diebold-Mariano"` test. Bandwidth for the long run variance estimator. If `NULL`, `bw` is selected according to `(3/4)*n^(1/3)`.
groups	Applicable to `"Ibragimov-Muller"` test. The number of groups to which the data is to be divided.
rhoLimit	Parameter `rhoLimit` limits to the absolute value of the estimated `rho` coefficient. This is useful as estimated values very close to 1 might cause instability.
...	Other parameters passed to algorithms.

Value

List containing loss differential estimate and associated p-value along with some other auxiliary information like the matrix of contrasts differentials `Phi` and the weights used for computation.

Examples

```
set.seed(1)
y <- rnorm(40)
m <- 36
h <- 1
v <- 1
algorithm1 <- function(y) {
  forecast::Arima(y, order = c(1, 0, 0))
}
algorithm2 <- function(y) {
  forecast::Arima(y, order = c(2, 0, 0))
}
testL(y, algorithm1, algorithm2, m = m, h = h, v = v)
```

`tsACV` *Perform time-series cross-validation*

Description

Function `tsACV()` computes contrasts between forecasts produced by a given algorithm and the original time-series on which the algorithm is trained. This can then be used to estimate the loss of the algorithm. Unlike the similar `tsCV()` function from the `'forecast'` package, `tsACV()` also records in-sample contrasts as these can be leveraged to produce more accurate out-of-sample loss estimates.

Usage

```
tsACV(
  y,
  algorithm,
  m,
  h = 1,
  v = 1,
  xreg = NULL,
  lossFunction = function(y, yhat) {     (y - yhat)^2 },
  ...
)
```

Arguments

`y`	Univariate time-series object.
`algorithm`	Algorithm which is to be applied to the time-series. The object which the algorithm produces should respond to `fitted` and `forecast` methods. Alternatively in the case of more complex custom algorithms, the algorithm may be a function which takes named arguments ("yInSample", "yOutSample", "h") or ("yInSample", "yOutSample", "h", "xregInSample", "xregOutSample") as inputs and

	produces a list with named elements ("yhatInSample", "yhatOutSample") containing vectors of in-sample and out-of-sample forecasts.
m	Length of the window on which the algorithm should be trained.
h	Number of predictions made after a single training of the algorithm.
v	Number of periods by which the estimation window progresses forward once the predictions are generated.
xreg	Matrix of exogenous regressors supplied to the algorithm (if applicable).
lossFunction	Loss function used to compute contrasts (defaults to squared error).
...	Other parameters passed to the algorithm.

Value

Matrix of computed contrasts Phi. Each row corresponds to a particular period of the y time-series and each column corresponds to a particular location of the training window.

Examples

```
set.seed(1)
y <- rnorm(40)
m <- 36
h <- 1
v <- 1
tsACV(y, forecast::Arima, m = m, h = h, v = v)
```

Index

estimateL, 2

testL, 3
tsACV, 5