Package ‘tbm’

April 9, 2025

Title Transformation Boosting Machines

Version 0.3-7

Date 2025-04-09

Description Boosting the likelihood of conditional and shift transformation models as introduced in <DOI:10.1007/s11222-019-09870-4>.

Depends mlt (>= 1.0-6), mboost (>= 2.8-2)

Imports variables, basefun, sandwich, coneproj, methods

Suggests TH.data (>= 1.0-9), tram (>= 0.2-3), survival, partykit, lattice, latticeExtra, knitr, colorspace, gamlss.data, trtf

VignetteBuilder knitr

URL `http://ctm.R-forge.R-project.org`

License GPL-2

NeedsCompilation no

Author Torsten Hothorn [aut, cre] (<`https://orcid.org/0000-0001-8301-0471`>)

Maintainer Torsten Hothorn <`Torsten.Hothorn@R-project.org`>

Repository CRAN

Date/Publication 2025-04-09 15:00:02 UTC

Contents

ctmboost *Likelihood Boosting for Conditional Transformation Models*

Description

Employs maximisation of the likelihood for estimation of conditional transformation models

Usage

```
ctmboost(model, formula, data = list(), weights = NULL,
         method = quote(mboost::mboost), ...)
```

Arguments

`model`	an object of class `mlt` as returned by `mlt`.
`formula`	a model formula describing how the parameters of `model` depend on explanatory variables, see `mboost`.
`data`	an optional data frame of observations.
`weights`	an optional vector of weights.
`method`	a call to `mboost`, `gamboost`, or `blackboost`.
`...`	additional arguments to `method`.

Details

The parameters of `model` depend on explanatory variables in a possibly structured additive way (see Hothorn, 2020). The number of boosting iterations is a hyperparameter which needs careful tuning.

Value

An object of class `ctmboost` with `predict` and `logLik` methods.

References

Torsten Hothorn (2020). Transformation Boosting Machines. *Statistics and Computing*, **30**, 141–152.

Examples

```
if (require("TH.data") && require("tram")) {
    data("bodyfat", package = "TH.data")

    ### estimate unconditional model
    m_mlt <- BoxCox(DEXfat ~ 1, data = bodyfat, prob = c(.1, .99))
    ### get corresponding in-sample log-likelihood
    logLik(m_mlt)

    ### estimate conditional transformation model
    bm <- ctmboost(m_mlt, formula = DEXfat ~ ., data = bodyfat,
                   method = quote(mboost::mboost))
    ### in-sample log-likelihood (NEEDS TUNING OF mstop!)
    logLik(bm)

    ### evaluate conditional densities for two observations
    predict(bm, newdata = bodyfat[1:2,], type = "density")
}
```

stmboost *Likelihood Boosting for Shift Transformation Models*

Description

Employs maximisation of the likelihood for estimation of shift transformation models

Usage

```
stmboost(model, formula, data = list(), weights = NULL,
         method = quote(mboost::mboost), mltargs = list(), ...)
```

Arguments

`model` an object of class `mlt` as returned by `mlt`.

`formula` a model formula describing how the parameters of `model` depend on explanatory variables, see `mboost`.

`data` an optional data frame of observations.

`weights` an optional vector of weights.

`method` a call to `mboost`, `gamboost`, or `blackboost`.

`mltargs` a list with arguments to be passed to `mlt`.

`...` additional arguments to `method`.

Details

The parameters of `model` depend on explanatory variables in a possibly structured additive way (see Hothorn, 2020). The number of boosting iterations is a hyperparameter which needs careful tuning.

Value

An object of class `stmboost` with `predict` and `logLik` methods.

References

Torsten Hothorn (2020). Transformation Boosting Machines. *Statistics and Computing*, **30**, 141–152.

Examples

```
if (require("TH.data") && require("tram")) {
    data("bodyfat", package = "TH.data")

    ### estimate unconditional model
    m_mlt <- BoxCox(DEXfat ~ 1, data = bodyfat, prob = c(.1, .99))
    ### get corresponding in-sample log-likelihood
    logLik(m_mlt)

    ### estimate conditional transformation model
    bm <- stmboost(m_mlt, formula = DEXfat ~ ., data = bodyfat,
                   method = quote(mboost::mboost))
    ### in-sample log-likelihood (NEEDS TUNING OF mstop!)
    logLik(bm)

    ### evaluate conditional densities for two observations
    predict(bm, newdata = bodyfat[1:2,], type = "density")
}
```

Index