Package 'BoostMLR'

March 15, 2020

Type Package

Title Boosting for Multivariate Longitudinal Response

Version 1.0.2

Date 2020-03-15

Author Amol Pande, Hemant Ishwaran

Maintainer Amol Pande <amoljpande@gmail.com>

Description Jointly models the multivariate longitudinal response and multiple covariates and time using gradient boosting approach.

License GPL (>= 2)

Depends R (>= 3.5.0)

Imports Rcpp (>= 0.12.18), stats, splines, nlme

Suggests mlbench

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-03-15 22:00:11 UTC

R topics documented:

BoostMLR-package 2
BoostMLR 3
BoostMLR.news 9
Laboratory_Data 10
partial.BoostMLR 10
plotBoostMLR 12
plotVIMP 14
predictBoostMLR 15
simLong 18
updateBoostMLR 19
vimp.BoostMLR 20

Index **23**

BoostMLR-package *Boosting for Multivariate Longitudinal Response*

Description

The primary feature of the package is to jointly model the multiple longitudinal responses (referred to as multivariate longitudinal response) and multiple covariates and time from a longitudinal study using gradient boosting approach (Pande et al., 2020). Covariates can be time-varying or time-invariant. Special cases include modeling of univariate longitudinal response from a longitudinal study, and univariate or multivariate response from a cross-sectional study. In all cases, responses are assumed to be continuous. The estimated coefficient can be a function of time (referred to as time-varying coefficient in case of a longitudinal study) or a function of pre-specified covariate (in case of a longitudinal or a cross-sectional study) or fixed.

Details

This package allows joint modeling of a multivariate longitudinal response, which is based on marginal model. Estimation is performed using gradient boosting, a generic form of boosting (Friedman J. H., 2001). The boosting approach use in this package is closely related to component-wise L2 boosting with L1 penalization. Package can handle high dimensionalilty of covariate and response when some of the covariates and responses are pure noise.

The package is designed to identify covariates that affect responses differently as different time intervals. This idea is helpful to dissect an overall effect of covariate into different time intervals. For example, some covariates affect response at the beginning of the follow-up whereas others at a later stage.

Package Overview

This package contains many useful functions and users should read the help file in its entirety for details. However, we briefly mention several key functions that may make it easier to navigate and understand the layout of the package.

1. `BoostMLR`

   This is the main entry point to the package. The model is fit using the gradient boosting approach for the user specified training data.

2. `updateBoostMLR` (`updateBoostMLR`)

   This allows to update the model by specifying additional boosting iteration.

3. `predictBoostMLR` (`predictBoostMLR`)

   Model performace can be obtained using the test set data. This function also estimate variable importance (VIMP).

Author(s)

Amol Pande and Hemant Ishwaran

Maintainer: Amol Pande <amoljpande@gmail.com>

References

Pande A., Ishwaran H., Blackstone E.H. (2020). Boosting for multivariate longitudinal response.

Friedman J.H. (2001). Greedy function approximation: a gradient boosting machine, *Ann. of Statist.*, 5:1189-1232.

See Also

BoostMLR, updateBoostMLR, predictBoostMLR, simLong

BoostMLR *Boosting for Multivariate Longitudinal Response*

Description

Function jointly models the multiple longitudinal responses (referred to as multivariate longitudinal response) and multiple covariates and time from a longitudinal study using gradient boosting approach (Pande et al., 2020). Covariates can be time-varying or time-invariant. Special cases include modeling of univariate longitudinal response from a longitudinal study, and univariate or multivariate response from a cross-sectional study. In all cases, responses are assumed to be continuous. The estimated coefficient can be a function of time (referred to as time-varying coefficient in case of a longitudinal study) or a function of pre-specified covariate (in case of a longitudinal or a cross-sectional study) or fixed.

Usage

```
BoostMLR(x,
         tm,
         id,
         y,
         Time_Varying = TRUE,
         BS_Time = TRUE,
         nknots_t = 10,
         d_t = 3,
         All_RawX = TRUE,
         RawX_Names,
         nknots_x = 7,
         d_x = 3,
         M = 200,
         nu = 0.05,
         Mod_Grad = TRUE,
         Shrink = FALSE,
         VarFlag = TRUE,
         lower_perc = 0.25,
         upper_perc = 0.75,
         NLambda = 100,
         Verbose = TRUE,
         ...)
```

Arguments

x	Data frame (or matrix) containing x-values (covariates). The number of rows should match with number of rows of response y. Covariates can be time-varying or time-invariant. Missing values are allowed, and are ignored during estimation. If unspecified, model will be fitted with time alone (applicable in the situation when the interest is to obtain an estimated mean response trajectory over time without the influence of any covariates).
tm	Vector of time values, one entry for each row of the response y. In case of a longitudinal study, the estimated coefficient will be a function of tm when Time_Varying = TRUE. If unspecified, data is assumed to be generated from a cross-sectional study, and the relationship between y and x can be obtained. In case of a longitudinal or cross-sectional study, coefficient can be a function of covariate z which is not a part of x by using z in place of tm or it can be fixed when Time_Varying = FALSE.
id	Vector of subject identifier with same length as the number of rows of y. If id is unspecified along with tm, data is assumed to be generated from a cross-sectional study, and the relationship between y and x can be obtained.
y	Data frame (or matrix) containing the y-values (response) in case of multivariate response or a vector of y-values in case of univariate response.
Time_Varying	Time-varying coefficient model or a fixed coefficient model?
BS_Time	If tm is specified, should tm is mapped using B-spline or use original scale of tm? Default is TRUE, which allows mapping of tm using B-spline.
nknots_t	If BS_Time = TRUE, specify number of knots for B-spline of tm.
d_t	If BS_Time = TRUE, specify degree of polynomial for B-spline of tm.
All_RawX	Use original scale of x or map each covariate using B-spline? Default is TRUE, which means original scale of x is used; if FALSE, covariates measured on continuous scale will be mapped using B-splines.
RawX_Names	If All_RawX = FALSE, specify names of the covariates, measured on a continuous scale, that should be used as it is without mapping using B-spline. Note that, even if All_RawX = FALSE, covariates not measured on a continuous scale, such as binary, nominal, and ordinal covariates will be used without mapping.
nknots_x	Specify number of knots for B-spline of x. This can be a vector of length equal to the number of covariates or a scalar. If scalar, same value will be used for all covariates.
d_x	Specify degree of polynomial for B-spline of x. This can be a vector of length equal to the number of covariates or a scalar. If scalar, same value will be used for all covariates.
M	Number of boosting iterations.
nu	Boosting regularization parameter. A value from the interval (0,1].
Mod_Grad	Use a modified gradient? Modified gradient is a special type of gradient that is independent of the correlation coefficient. Pande A. (2017) observed that prediction performance increases under modified gradient.
Shrink	Allow estimated coefficient to shrink to zero using L1 penalization?

VarFlag	Estimate the variance (scale parameter) and correlation parameter for each y? Applicable for a longitudinal study. If VarFlag = FALSE, a fixed value of scale parameter = 1 and correlation parameter = 0 is used.
lower_perc	Lower percentile value is used to determine the lower cut-off for the distribution of parameter estimate. Applicable when Shrink = TRUE. Refer to Pande et al. (2020) for details.
upper_perc	Upper percentile value is used to determine the upper cut-off for the distribution of parameter estimate. Applicable when Shrink = TRUE. Refer to Pande et al. (2020) for details.
NLambda	Number of replications for generating distribution of parameter estimates. Applicable when Shrink = TRUE. Refer to Pande et al. (2020) for details.
Verbose	Print the current stage of boosting iteration?
...	Further arguments passed to or from other methods.

Details

This is a non-parametric approach for joint modeling of a multivariate longitudinal response, which is based on marginal model. Estimation is performed using gradient boosting, a generic form of boosting (Friedman J. H., 2001). Our boosting approach is closely related to component-wise L2 boosting with L1 penalization. Approach can handle high dimensionalilty of covariate and response when some of the covariates and responses are pure noise.

Approach is designed to identify covariates that affect responses differently as different time intervals. This idea is helpful to dissect an overall effect of covariate into different time intervals. For example, some covariates affect response at the beginning of the follow-up whereas others at a later stage.

Shrinking allows for early termination of boosting to prevent overfitting. Also, it provides a parsimonious model by shrinking coefficient for non-informative covariate-response pair to zero.

Value

x	Matrix containing x-values.
id	Vector of subject identifier.
tm	Vector of time values.
y	Matrix containing y-values.
UseRaw	Logical vector indicating indexes of covariates which are used as it is without B-spline mapping.
x_Names	Variable names of x.
y_Names	Variable names of y.
M	Number of boosting iterations. If boosting terminates before a pre-specified M, this indicates the last boosting iteration before termination.
nu	Regularization parameter.
Tm_Beta	An estimate of the parameter beta. This consist of a list of length equal to the number of multivariate response (denoted by L). If Time_Varying = TRUE, each element from the list represents a matrix with number of columns equal

to the number of covariates and the number of rows equal to the length of `tm`. Each column of the matrix represents an estimate of time-varying coefficient for the given covariate. If `Time_Varying` = FALSE, in place of estimate of time-varying coefficient, a fixed estimate is provided similar to the estimate from a parametric model. The result is provided for covariates who are treated as it is (i.e., the original scale); for covariates who are mapped using B-spline, the estimates are difficult to interprete and therefore the output is NA.

`mu` Estimate of the conditional expectation of `y` corresponding to the M'th boosting iterations.

`Error_Rate` Training error rate for each response across the boosting iterations.

`Variable_Select`
Indexes of important covariates that get picked-up across time and across boosting iterations. Result is shown as a matrix with M rows and H (number of overlapping time intervals) columns, where each element represents index of covariate.

`Response_Select`
Indexes of important responses that get picked-up across time and across boosting iterations. Result is shown as a matrix with M rows and H columns, where each element represents index of response variable.

`VarFlag` Whether the variance (scale parameter) and correlation are estimated?

`Time_Varying` Whether estimates are time-varying or fixed?

`Phi` Matrix, having dimension M by L, representing an estimate of variance (scale parameter) for each response across the boosting iterations.

`Rho` Matrix, having dimension M by L, represent an estimate of correlation for each response across the boosting iterations.

`Lambda_List` Estimate of the lambda (the L1 penaulty parameter) for each boosting iterations. Useful for internal calculation.

`Grow_Object` Useful for internal calculation.

Author(s)

Amol Pande and Hemant Ishwaran

References

Pande A., Ishwaran H., Blackstone E.H. (2020). Boosting for multivariate longitudinal response.

Pande A., Li L., Rajeswaran J., Ehrlinger J., Kogalur U.B., Blackstone E.H., Ishwaran H. (2017). Boosted multivariate trees for longitudinal data, *Machine Learning*, 106(2): 277–305.

Pande A. (2017). *Boosting for longitudinal data*. Ph.D. Dissertation, Miller School of Medicine, University of Miami.

Friedman J.H. (2001). Greedy function approximation: a gradient boosting machine, *Ann. of Statist.*, 5:1189-1232.

See Also

`updateBoostMLR`, `predictBoostMLR`, `simLong`

Examples

```
##------------------------------------------------------------------
## Multivariate Longitudinal Response
##------------------------------------------------------------------

# Simulate data involves 3 response and 4 covariates

dta <- simLong(n = 100, N = 5, rho =.80, model = 1, q_x = 0,
                                  q_y = 0,type = "corCompSym")$dtaL

# Boosting call: Raw values of covariates, B-spline for time,
# no shrinkage, no estimate of rho and phi

boost.grow <- BoostMLR(x = dta$features, tm = dta$time, id = dta$id,
                           y = dta$y, M = 100, VarFlag = FALSE)

# Plot training error
plotBoostMLR(boost.grow$Error_Rate,xlab = "m",ylab = "Training Error")


##------------------------------------------------------------------
## Laboratory data
##------------------------------------------------------------------

data(Laboratory_Data, package = "BoostMLR")

Var_Names <- colnames(Laboratory_Data)
x_Names <- setdiff(Var_Names, c("id","time","tbili_po","creat_po"))

dta_id <- Laboratory_Data[,"id"]
dta_time <- Laboratory_Data[,"time"]
dta_x <- Laboratory_Data[,x_Names]
dta_y <- Laboratory_Data[,c("tbili_po","creat_po")]

boost.grow <- BoostMLR(x = dta_x,tm = dta_time,id = dta_id,y = dta_y,
                           Time_Varying = TRUE,BS_Time = TRUE,
                           All_RawX = TRUE,M = 10, VarFlag = TRUE)

##------------------------------------------------------------------
## Univariate Longitudinal Response
##------------------------------------------------------------------

# Simulate data involves 1 response and 4 covariates

dta <- simLong(n = 100, N = 5, rho =.80, model = 2, q_x = 0,
                                  q_y = 0,type = "corCompSym")$dtaL

# Boosting call: B-spline for time and covariates, shrinkage,
# estimate of rho and phi
```

```
boost.grow <- BoostMLR(x = dta$features, tm = dta$time, id = dta$id,
                         y = dta$y, M = 100, BS_Time = TRUE,
                         All_RawX = FALSE, Shrink = TRUE,VarFlag = TRUE)

# Plot training error
plotBoostMLR(boost.grow$Error_Rate,xlab = "m",ylab = "Training Error")

# Plot phi
plotBoostMLR(boost.grow$Phi,xlab = "m",ylab = "phi")

# Plot rho
plotBoostMLR(boost.grow$Rho,xlab = "m",ylab = "rho")


##-------------------------------------------------------------------
## Multivariate Longitudinal Response
##-------------------------------------------------------------------

# Simulate data involves 3 response and 4 covariates

dta <- simLong(n = 100, N = 5, rho =.80, model = 1, q_x = 0,
                                  q_y = 0,type = "corCompSym")$dtaL

# Boosting call: Raw values of covariates, fixed parameter estimates
# instead of time varying, no shrinkage, no estimate of rho and phi

boost.grow <- BoostMLR(x = dta$features, tm = dta$time, id = dta$id,
                       y = dta$y, M = 100,Time_Varying = FALSE,VarFlag = FALSE)

# Print parameter estimates
boost.grow$Tm_Beta

##-------------------------------------------------------------------
## Multivariate Response from Cross-sectional Data: Estimated
## coefficient as a function of covariate
##-------------------------------------------------------------------

if (library("mlbench", logical.return = TRUE)) {
data("BostonHousing")

x <- BostonHousing[,c(1:7,9:12)]
tm <- BostonHousing[,8]
id <- 1:nrow(BostonHousing)
y <- BostonHousing[,13:14]

# Boosting call: Raw values of covariates, B-spline for covariate "dis",
# no shrinkage

boost.grow <- BoostMLR(x = x, tm = tm, id = id, y = y, M = 100,VarFlag = FALSE)

# Plot training error
plotBoostMLR(boost.grow$Error_Rate,xlab = "m",ylab = "Training Error",
                                            legend_fraction_x = 0.2)
```

```
}
##-------------------------------------------------------------------
## Univariate Response from Cross-sectional Data: Fixed estimated
## coefficient
##-------------------------------------------------------------------

if (library("mlbench", logical.return = TRUE)) {
library(mlbench)
data("BostonHousing")

x <- BostonHousing[,1:13]
y <- BostonHousing[,14]

# Boosting call: Raw values of covariates

boost.grow <- BoostMLR(x = x, y = y, M = 100)

# Plot training error
plotBoostMLR(boost.grow$Error_Rate,xlab = "m",ylab = "Training Error",
                                                    legend_fraction_x = 0.2)
}
```

`BoostMLR.news` *Show the NEWS file*

Description

Show the NEWS file of the BoostMLR package.

Usage

```
BoostMLR.news(...)
```

Arguments

`...`	Further arguments passed to or from other methods.

Value

None.

Author(s)

Amol Pande and Hemant Ishwaran

Laboratory_Data *Laboratory Data*

Description

The laboratory data is based on 459 patients who were listed for heart transplant and were put on mechanical circulatory system through device implantation from December 1991 to July 2009 at Cleveland Clinic. These patients had periodic measurements of their bilirubin and creatinine levels. Data from 459 patients includes 18285 measurements of bilirubin and creatinine with an average of 39 measurements per patient.

Format

Laboratory data has 4 parts:

1. A total of 41 x-variables.
2. Time points (time).
3. Patient identifier (id).
4. Longitudinal responses (tbili_po and creat_po).

References

Rajeswaran J., Blackstone E.H. and Bernard J. Evolution of association between renal and liver function while awaiting for the heart transplant: An application using bivariate multiphase nonlinear mixed effect model. *Statistical methods in medical research* 27(7):2216–2230, 2018.

Examples

```
data(Laboratory_Data, package = "BoostMLR")
```

partial.BoostMLR *Partial plot analysis*

Description

Partial dependence plot of x and time against adjusted predicted y.

Usage

```
## S3 method for class 'BoostMLR'
partial(Object,
        xvar.name,
        n.x = 10,
        n.tm = 10,
        x.unq = NULL,
        tm.unq = NULL,
        Mopt,
        plot.it = TRUE,
        path_saveplot = NULL,
        Verbose = TRUE,
        ...)
```

Arguments

`Object`	A boosting object of class `(BoostMLR,grow)`.
`xvar.name`	Name of the x-variable to be used for partial plot.
`n.x`	Maximum number of unique points used for `xvar.name`. Reduce this value if plotting is slow.
`n.tm`	Maximum number of unique points used for `tm`. Reduce this value if plotting is slow.
`x.unq`	Unique values used for the partial plot for variable `xvar.name`. Default is NULL in which case unique values are obtained uniformaly based on the range of variable.
`tm.unq`	Unique time points used for the partial plots of x against y. Default is NULL in which case unique values are obtained uniformaly based on the range of `tm`.
`Mopt`	The optimal number of boosting iteration. If missing, the value from the `Object` will be used.
`plot.it`	Should partial plot be displayed?
`path_saveplot`	Provide the location where plot should be saved. By default the plot will be saved at temporary folder.
`Verbose`	Display the path where the plot is saved?
`...`	Further arguments passed to or from other methods.

Details

Partial dependence plot (Friedman, 2001) of x values specified by `xvar.name` against the adjusted predicted y-values over a set of time points specified by `tm.unq`.

Value

`x.unq`	Unique values used for the partial plot for variable `xvar.name`
`tm.unq`	Unique time points used for the partial plots of x against y.

pList — List with number of elements equal to number of multivariate response. Each element of the list is a matrix with number of rows equal to length of x.unq, and number of columns equal to length of tm.unq. Values in the matrix represent predicted partial values.

sList — List with number of elements equal to number of multivariate response. Each element is a matrix with the same dimension as described in pList. Values are calculated using the local smoother (loess) for tm.unq and the i'th row of the matrix from pList. Users are encouraged to use pList to genenrate their own sList so that they will have more control over the different arguments of local smoother.

Author(s)

Amol Pande and Hemant Ishwaran

References

Friedman J.H. Greedy function approximation: a gradient boosting machine, *Ann. of Statist.*, 5:1189-1232, 2001.

Examples

```
##------------------------------------------------------------
## Generate partial plot for covariate x1
##------------------------------------------------------------

dta <- simLong(n = 100, N = 5, rho =.80, model = 1, q_x = 0,
                                  q_y = 0,type = "corCompSym")$dtaL

# Boosting call: Raw values of covariates, B-spline for time,
# no shrinkage, no estimate of rho and phi

boost.grow <- BoostMLR(x = dta$features, tm = dta$time, id = dta$id,
                           y = dta$y, M = 100, VarFlag = FALSE)

Partial_Plot_x1 <- partial.BoostMLR(Object = boost.grow, xvar.name = "x1",plot.it = FALSE)
```

---

plotBoostMLR *Plotting results across across the boosting iterations.*

Description

Plotting training and test error, and estimate of variance/correlation parameters across the boosting iterations.

Usage

```
plotBoostMLR(Result,
             xlab = "",
             ylab = "",
             legend_fraction_x = 0.10,
             legend_fraction_y = 0,
             ...)
```

Arguments

Result	Result in the matrix form either training or test error, or estimate of variance/correlation parameters across the boosting iterations.
xlab	Label for the x-axis.
ylab	Label for the y-axis.
legend_fraction_x	Value use to expland the x-axis.
legend_fraction_y	Value use to expland the y-axis.
...	Further arguments passed to or from other methods.

Details

Plotting training and test error, and estimate of variance/correlation parameters across the boosting iterations.

Author(s)

Amol Pande and Hemant Ishwaran

Examples

```
##------------------------------------------------------------------
## Multivariate Longitudinal Response
##------------------------------------------------------------------

# Simulate data involves 3 response and 4 covariates

dta <- simLong(n = 100, N = 5, rho =.80, model = 1, q_x = 0,
                                  q_y = 0,type = "corCompSym")$dtaL

# Boosting call: Raw values of covariates, B-spline for time,
# no shrinkage, no estimate of rho and phi

boost.grow <- BoostMLR(x = dta$features, tm = dta$time, id = dta$id,
                           y = dta$y, M = 100, VarFlag = FALSE)

# Plot training error
plotBoostMLR(boost.grow$Error_Rate,xlab = "m",ylab = "Training Error")
```

plotVIMP *Variable Importance (VIMP) plot*

Description

Barplot displaying variable importance for the main effect.

Usage

```
plotVIMP(vimp_Object,
         xvar.names = NULL,
         cex.xlab = NULL,
         ymaxlim = 0,
         yminlim = 0,
         main = "Variable Importance (%)",
         col = grey(0.8),
         cex.lab = 1.5,
         ylbl = NULL,
         legend_placement = NULL,
         plot.it = TRUE,
         path_saveplot = NULL,
         Verbose = TRUE)
```

Arguments

vimp_Object	List with number of elements equal to the number of response variables.
xvar.names	Names of the covariates. If NULL, names will be pulled from vimp_Object.
cex.xlab	Magnification of the names of the covariates for the barplot.
ymaxlim	By default, we use the range of the vimp values for the barplot limit on the y-axis. If one wants to extend the limit, add the amount with which the limit will extend above the x-axis.
yminlim	Similar to ymaxlim, this will add the amount with which the limit will extend below the x-axis.
main	Main title for the plot.
col	Color of the plot.
cex.lab	Magnification of the x and y lables.
ylbl	Label for the y-axis.
legend_placement	Do you want name of the covariates on top of the each barplot? If so, use default setting; else set value on the negative direction of y-axis which arrange covariate name beneath the barplot.
plot.it	Should the VIMP plot be displayed?
path_saveplot	Provide the location where plot should be saved. By default the plot will be saved at temporary folder.
Verbose	Display the path where the plot is saved?

Details

Barplot displaying VIMP for each response. Barplot will be save as pdf file in the working directory.

Author(s)

Amol Pande and Hemant Ishwaran

Examples

```
##-------------------------------------------------------------------
## VIMP plot for multivariate longitudinal response
##-------------------------------------------------------------------

# Simulate data involves 3 response and 4 covariates

dta <- simLong(n = 100, ntest = 100 ,N = 5, rho =.80, model = 1, q_x = 0,
                                    q_y = 0,type = "corCompSym")
dtaL <- dta$dtaL
trn <- dta$trn
# Boosting call: Raw values of covariates, B-spline for time,
# no shrinkage, no estimate of rho and phi

boost.grow <- BoostMLR(x = dtaL$features[trn,], tm = dtaL$time[trn],
                       id = dtaL$id[trn], y = dtaL$y[trn,], M = 100, VarFlag = FALSE)

boost.pred <- predictBoostMLR(Object = boost.grow, x = dtaL$features[-trn,],
                              tm = dtaL$time[-trn], id = dtaL$id[-trn],
                              y = dtaL$y[-trn,], importance = TRUE)

# Plot VIMP
plotVIMP(vimp_Object = boost.pred$vimp,ymaxlim = 20,plot.it = FALSE)
```

`predictBoostMLR` *Prediction for the multivariate longitudinal response*

Description

Function returns predicted values for the response. Also, if the response is provided, function returns test set performance, optimal boosting iteration, and variable importance (VIMP).

Usage

```
predictBoostMLR(Object,
                x,
                tm,
                id,
                y,
```

```
M,
importance = FALSE,
eps = 1e-5,
...)
```

Arguments

Object	A boosting object obtained using the function BoostMLR on the training data.
x	Data frame (or matrix) containing the test set x-values (covariates). Covariates can be time-varying or time-invariant. If x is unspecified while growing the Object, it should be unspecified here as well.
tm	Vector of test set time values. If tm is unspecified while growing the Object, it should be unspecified here as well.
id	Vector of test set subject identifier. If id is unspecified while growing the Object, it should be unspecified here as well.
y	Data frame (or matrix) containing the test set y-values (response) in case of multivariate response or a vector of y-values in case of univariate response. If y is unspecified then predicted values corresponding to x and tm can be obtained but no performance measure such as test set error and VIMP.
M	Number of boosting iterations. Value should be less than or equal to the value specified in the Object. If unspecified, value from the Object will be used.
importance	Whether to calculate standardized variable importance (VIMP) for each covariate?
eps	Tolerance value used for determining the optimal M.
...	Further arguments passed to or from other methods.

Details

The predicted response and performance values are obtained for the test data using the Object grown using function BoostMLR on the training data.

Value

Data	A list with elements x, tm, id and y. Additionally, the list include mean and standard deviation of x and y.
x_Names	Variable names of x.
y_Names	Variable names of y.
mu	Estimate of conditional expectation of y corresponding to the last boosting iteration.
mu_Mopt	Estimate of conditional expectation of y corresponding to the optimal boosting iteration.
Error_Rate	Test set error rate for each multivariate response across the boosting iterations.
Mopt	The optimal number of boosting iteration.
nu	Regularization parameter.

rmse	Test set standardized root mean square error (sRMSE) at the Mopt.
vimp	Standardized VIMP for each covariate. This consist of a list of length equal to the number of multivariate response. Each element from the list represents a matrix with number of rows equal to the number of covariates and the number of columns equal to the number of overlapping time intervals + 1 where the first column contains covariate main effects and all other columns contain covariate-time interaction effects.
Pred_Object	Useful for internal calculation.

Author(s)

Amol Pande and Hemant Ishwaran

References

Pande A., Ishwaran H., Blackstone E.H. (2020). Boosting for multivariate longitudinal response.

Pande A., Li L., Rajeswaran J., Ehrlinger J., Kogalur U.B., Blackstone E.H., Ishwaran H. (2017). Boosted multivariate trees for longitudinal data, *Machine Learning*, 106(2): 277–305.

Pande A. (2017). *Boosting for longitudinal data*. Ph.D. Dissertation, Miller School of Medicine, University of Miami.

See Also

BoostMLR, updateBoostMLR, simLong

Examples

```
##------------------------------------------------------------------
## Multivariate Longitudinal Response
##------------------------------------------------------------------

# Simulate data involves 3 response and 4 covariates

dta <- simLong(n = 100, ntest = 100 ,N = 5, rho =.80, model = 1, q_x = 0,
                                  q_y = 0,type = "corCompSym")
dtaL <- dta$dtaL
trn <- dta$trn
# Boosting call: Raw values of covariates, B-spline for time,
# no shrinkage, no estimate of rho and phi

boost.grow <- BoostMLR(x = dtaL$features[trn,], tm = dtaL$time[trn],
                      id = dtaL$id[trn], y = dtaL$y[trn,], M = 100, VarFlag = FALSE)

boost.pred <- predictBoostMLR(Object = boost.grow, x = dtaL$features[-trn,],
                              tm = dtaL$time[-trn], id = dtaL$id[-trn],
                              y = dtaL$y[-trn,], importance = TRUE)
# Plot test set error
plotBoostMLR(boost.pred$Error_Rate,xlab = "m",ylab = "Test Set Error",
                                            legend_fraction_x = 0.2)
```

simLong *Simulate longitudinal data*

Description

Simulates longitudinal data from multivariate and univariate longitudinal response model.

Usage

```
simLong(n = 100,
        ntest = 0,
        N = 5,
        rho = 0.8,
        model = c(1, 2),
        phi = 1,
        q_x = 0,
        q_y = 0,
        type = c("corCompSym", "corAR1", "corSymm", "iid"))
```

Arguments

n	Requested training sample size.
ntest	Requested test sample size.
N	Parameter controlling number of time points per subject.
rho	Correlation parameter.
model	Requested simulation model.
phi	Variance of measurement error.
q_x	Number of noise covariates.
q_y	Number of noise responses.
type	Type of correlation matrix.

Details

Simulates longitudinal data from multivariate and univariate longitudinal response model. We consider following 2 models:

1. model=1*:* Simpler linear model consist of three longitudinal responses, y1, y2, and y3 and four covariates x1, x2, x3, and x4. Response y1 is associated with x1 and x4. Response y2 is associated with x2 and x4. Response y3 is associated with x3 and x4.
2. model=2*:* Relatively complex model consist of single longitudinal response and four covariates. Model includes non-linear relationship between response and covariates and covariate-time interaction.

Value

An invisible list with the following components:

dtaL	List containing the simulated data in the following order: features, time, id and y.
dta	Simulated data given as a data frame.
trn	Index of id values identifying the training data.

Author(s)

Amol Pande and Hemant Ishwaran

References

Pande A., Li L., Rajeswaran J., Ehrlinger J., Kogalur U.B., Blackstone E.H., Ishwaran H. (2017). Boosted multivariate trees for longitudinal data, *Machine Learning*, 106(2): 277–305.

---

updateBoostMLR *Update boosting object with an additional boosting iterations*

Description

Function allows to update boosting object with an additional boosting iterations.

Usage

```
updateBoostMLR(Object,
               M_Add,
               Verbose = TRUE,
               ...)
```

Arguments

Object	Boosting object. This object is previously obtained using BoostMLR function or using update function.
M_Add	Number of additional boosting iterations.
Verbose	Print the current stage of boosting iteration?
...	Further arguments passed to or from other methods.

Details

In boosting, Mopt, the number of boosting iterations required to achive optimal result, is unknown. Typically, Mopt is estimated by specifying a large value of M and then search for an optimal value that is less than M using the test data. Function update allows user to start with a small value of M, and keep increamenting boosting iterations, each time running through the test data, until an optimal boosting iteration is found. This can significantly reduce unnecessary computations, particularly when Mopt « M. The procedure can be replicated multiple times using the boosting object (see example below). Results from update can be treated the same way we treat results from BoostMLR.

Author(s)

Amol Pande and Hemant Ishwaran

See Also

BoostMLR, predictBoostMLR, simLong

Examples

```
##-------------------------------------------------------------------
## Univariate Longitudinal Response
##-------------------------------------------------------------------

# Simulate data involves 1 response and 4 covariates

dta <- simLong(n = 100, N = 5, rho =.80, model = 2, q_x = 0,
                                  q_y = 0,type = "corCompSym")$dtaL

# Boosting call: Raw values of covariates, B-spline for time,
# no shrinkage, no estimate of rho and phi

boost.grow <- BoostMLR(x = dta$features, tm = dta$time, id = dta$id,
                           y = dta$y, M = 100, VarFlag = FALSE)

# Update boosting object for the additional 100 iteration
boost.grow <- updateBoostMLR(Object = boost.grow, M_Add = 100,Verbose = TRUE)

# Update boosting object for the additional 50 iteration
boost.grow <- updateBoostMLR(Object = boost.grow, M_Add = 50,Verbose = TRUE)
```

---

vimp.BoostMLR *Variable Importance*

Description

Calculate standardized variable importance (VIMP) for each covariate or a joint VIMP of multiple covariates.

Usage

```
## S3 method for class 'BoostMLR'
vimp(Object,
     xvar.names = NULL,
     joint = FALSE)
```

Arguments

Object	A boosting object of class (BoostMLR,predict).
xvar.names	Names of the x-variables for which VIMP is requested. If NULL, VIMP is calcuated for all the covariates.
joint	Whether to estimate VIMP for each covariate from xvar.names or a joint VIMP for multiple covariates?

Details

Standardized variable importance (VIMP) is calcuated for each covariate or a joint VIMP is calculated for all the covariates specified in xvar.names.

Value

If joint = FALSE, a standardized VIMP for each covariate is obtained otherwisea joint VIMP for all the covariates is obtained. The result consists of a list of length equal to the number of multivariate response. Each element from the list represents a matrix with number of rows equal to the number of covariates (in case of joint VIMP, the matrix will have a single row) and the number of columns equal to the number of overlapping time intervals + 1 where the first column contains covariate main effects and all other columns contain covariate-time interaction effects.

Author(s)

Amol Pande and Hemant Ishwaran

References

Pande A., Ishwaran H., Blackstone E.H. (2020). Boosting for multivariate longitudinal response.

Friedman J.H. Greedy function approximation: a gradient boosting machine, *Ann. of Statist.*, 5:1189-1232, 2001.

Examples

```
##------------------------------------------------------------------
## Calculate individual and joint VIMP
##------------------------------------------------------------------

# Simulate data involves 3 response and 4 covariates

dta <- simLong(n = 100, ntest = 100 ,N = 5, rho =.80, model = 1, q_x = 0,
                                    q_y = 0,type = "corCompSym")
```

```
dtaL <- dta$dtaL
trn <- dta$trn
# Boosting call: Raw values of covariates, B-spline for time,
# no shrinkage, no estimate of rho and phi

boost.grow <- BoostMLR(x = dtaL$features[trn,], tm = dtaL$time[trn],
                       id = dtaL$id[trn], y = dtaL$y[trn,], M = 100, VarFlag = FALSE)

boost.pred <- predictBoostMLR(Object = boost.grow, x = dtaL$features[-trn,],
                              tm = dtaL$time[-trn], id = dtaL$id[-trn],
                              y = dtaL$y[-trn,], importance = FALSE)
# Individual VIMP
Ind_vimp <- vimp.BoostMLR(boost.pred)

# Joint VIMP
Joint_vimp <- vimp.BoostMLR(boost.pred,joint = TRUE)
```

Index

∗Topic **boosting**
BoostMLR, 3
predictBoostMLR, 15
updateBoostMLR, 19
∗Topic **datasets**
Laboratory_Data, 10
∗Topic **documentation**
BoostMLR.news, 9
∗Topic **package**
BoostMLR-package, 2
∗Topic **plot**
partial.BoostMLR, 10
plotBoostMLR, 12
plotVIMP, 14
vimp.BoostMLR, 20
∗Topic **simulation**
simLong, 18
∗Topic **variable selection**
simLong, 18

BoostMLR, *2, 3*, 3, *17*, *20*
BoostMLR-package, 2
BoostMLR.news, 9

Laboratory_Data, 10

partial.BoostMLR, 10
plotBoostMLR, 12
plotVIMP, 14
predictBoostMLR, *2, 3*, *6*, 15, *20*

simLong, *3*, *6*, *17*, 18, *20*

updateBoostMLR, *2, 3*, *6*, *17*, 19

vimp.BoostMLR, 20