Package 'bartBMA'

October 12, 2022

Type Package

Title Bayesian Additive Regression Trees using Bayesian Model Averaging

Version 1.0

Date 2020-02-05

Author Belinda Hernandez [aut, cre]
Adrian E. Raftery [aut]
Stephen R Pennington [aut]
Andrew C. Parnell [aut]
Eoghan O'Neill [ctb]

Maintainer Belinda Hernandez <HERNANDB@tcd.ie>

Description

``BART-BMA Bayesian Additive Regression Trees using Bayesian Model Averaging" (Hernandez B, Raftery A.E., Parnell A.C. (2018) <doi:10.1007/s11222-017-9767-1>) is an extension to the original BART sum-of-trees model (Chipman et al 2010). BART-BMA differs to the original BART model in two main aspects in order to implement a greedy model which will be computationally feasible for high dimensional data. Firstly BART-BMA uses a greedy search for the best split points and variables when growing decision trees within each sum-of-trees model. This means trees are only grown based on the most predictive set of split rules. Also rather than using Markov chain Monte Carlo (MCMC), BART-BMA uses a greedy implementation of Bayesian Model Averaging called Occam's Window which take a weighted average over multiple sum-of-trees models to form its overall prediction. This means that only the set of sum-of-trees for which there is high support from the data are saved to memory and used in the final model.

License GPL (>= 2)

Imports Rcpp (>= 1.0.0), mvnfast, Rdpack

RdMacros Rdpack

LinkingTo Rcpp, RcppArmadillo, BH

RoxygenNote 7.0.2

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-03-13 11:50:05 UTC

R topics documented:

`bartBMA` *Bayesian Additive Regression Trees Using Bayesian Model Averaging (BART-BMA)*

Description

This is an implementation of Bayesian Additive Regression Trees (Chipman et al. 2010) using Bayesian Model Averaging (Hernandez et al. 2018).

Usage

```
bartBMA(x.train, ...)

## Default S3 method:
bartBMA(
  x.train,
  y.train,
  a = 3,
  nu = 3,
  sigquant = 0.9,
```

```
  c = 1000,
  pen = 12,
  num_cp = 20,
  x.test = matrix(0, 0, 0),
  num_rounds = 5,
  alpha = 0.95,
  beta = 2,
  split_rule_node = 0,
  gridpoint = 0,
  maxOWsize = 100,
  num_splits = 5,
  gridsize = 10,
  zero_split = 1,
  only_max_num_trees = 1,
  min_num_obs_for_split = 2,
  min_num_obs_after_split = 2,
  exact_residuals = 1,
  spike_tree = 0,
  s_t_hyperprior = 1,
  p_s_t = 0.5,
  a_s_t = 1,
  b_s_t = 3,
  lambda_poisson = 10,
  less_greedy = 0,
  ...
)
```

Arguments

x.train	Training data covariate matrix
...	Further arguments.
y.train	Training data outcome vector.
a	This is a parameter that influences the variance of terminal node parameter values. Default value a=3.
nu	This is a hyperparameter in the distribution of the variance of the error term. THe inverse of the variance is distributed as Gamma (nu/2, nu*lambda/2). Default value nu=3.
sigquant	Calibration quantile for the inverse chi-squared prior on the variance of the error term.
c	This determines the size of Occam's Window
pen	This is a parameter used by the Pruned Exact Linear Time Algorithm when finding changepoints. Default value pen=12.
num_cp	This is a number between 0 and 100 that determines the proportion of changepoints proposed by the changepoint detection algorithm to keep when growing trees. Default num_cp=20.
x.test	Test data covariate matrix. Default x.test=matrix(0.0,0,0).

num_rounds Number of trees. (Maximum number of trees in a sum-of-tree model). Default num_rounds=5.

alpha Parameter in prior probability of tree node splitting. Default alpha=0.95

beta Parameter in prior probability of tree node splitting. Default beta=1

split_rule_node
Binary variable. If equals 1, then find a new set of potential splitting points via a changepoint algorithm after adding each split to a tree. If equals zero, use the same set of potential split points for all splits in a tree. Default split_rule_node=0.

gridpoint Binary variable. If equals 1, then a grid search changepoint detection algorithm will be used. If equals 0, then the Pruned Exact Linear Time (PELT) changepoint detection algorithm will be used (Killick et al. 2012). Default gridpoint=0.

maxOWsize Maximum number of models to keep in Occam's window. Default maxOWsize=100.

num_splits Maximum number of splits in a tree

gridsize This integer determines the size of the grid across which to search if gridpoint=1 when finding changepoints for constructing trees.

zero_split Binary variable. If equals 1, then zero split trees can be included in a sum-of-trees model. If equals zero, then only trees with at least one split can be included in a sum-of-trees model.

only_max_num_trees
Binary variable. If equals 1, then only sum-of-trees models containing the maximum number of trees, num_rounds, are selected. If equals 0, then sum-of-trees models containing less than num_rounds trees can be selected. The default is only_max_num_trees=1.

min_num_obs_for_split
This integer determines the minimum number of observations in a (parent) tree node for the algorithm to consider potential splits of the node.

min_num_obs_after_split
This integer determines the minimum number of observations in a child node resulting from a split in order for a split to occur. If the left or right chikd node has less than this number of observations, then the split can not occur.

exact_residuals
Binary variable. If equal to 1, then trees are added to sum-of-tree models within each round of the algorithm by detecting changepoints in the exact residuals. If equals zero, then changepoints are detected in residuals that are constructed from approximate predictions.

spike_tree If equal to 1, then the Spike-and-Tree prior will be used, otherwise the standard BART prior will be used. The number of splitting variables has a beta-binomial prior. The number of terminal nodes has a truncated Poisson prior, and then a uniform prior is placed on the set of valid constructions of trees given the splitting variables and number of terminal nodes.

s_t_hyperprior If equals 1 and spike_tree equals 1, then a beta distribution hyperprior is placed on the variable inclusion probabilities for the spike and tree prior. The hyperprior parameters are a_s_t and b_s_t.

p_s_t	If spike_tree=1 and s_t_hyperprior=0, then p_s_t is the prior variable inclusion probability.
a_s_t	If spike_tree=1 and s_t_hyperprior=1, then a_s_t is a parameter of a beta distribution hyperprior.
b_s_t	If spike_tree=1 and s_t_hyperprior=1, then b_s_t is a parameter of a beta distribution hyperprior.
lambda_poisson	This is a parameter for the Spike-and-Tree prior. It is the parameter for the (truncated and conditional on the number of splitting variables) Poisson prior on the number of terminal nodes.
less_greedy	If equal to one, then a less greedy model search algorithm is used.

Value

The following objects are returned by bartbma:

fitted.values	The vector of predictions of the outcome for all training observations.
sumoftrees	This is a list of lists of matrices. The outer list corresponds to a list of sum-of-tree models, and each element of the outer list is a list of matrices describing the structure of the trees within a sum-of-tree model. See details.
obs_to_termNodesMatrix	This is a list of lists of matrices. The outer list corresponds to a list of sum-of-tree models, and each element of the outer list is a list of matrices describing to which node each of the observations is allocated to at all depths of each tree within a sum-of-tree model. See details.
bic	This is a vector of BICs for each sum-of-tree model.
test.preds	A vector of test data predictions. This output only is given if there is test data in the input.
sum_residuals	CURRENTLY INCORRECT OUTPUT. A List (over sum-of-tree models) of lists (over single trees in a model) of vectors of partial residuals. Unless the maximum number of trees in a model is one, in which case the output is a list (over single tree models) of vectors of partial residuals, which are all equal to the outcome vector.
numvars	This is the total number of variables in the input training data matrix.
call	match.call returns a call in which all of the specified arguments are specified by their full names.
y_minmax	Range of the input training data outcome vector.
response	Input taining data outcome vector.
nrowTrain	number of observations in the input training data.
sigma	sd(y.train)/(max(y.train)-min(y.train))
a	input parameter
nu	input parameter
lambda	parameter determined by the inputs sigma, sigquant, and nu

References

Chipman HA, George EI, McCulloch RE, others (2010). "BART: Bayesian additive regression trees." *The Annals of Applied Statistics*, **4**(1), 266–298.

Hernandez B, Raftery AE, Pennington SR, Parnell AC (2018). "Bayesian additive regression trees using Bayesian model averaging." *Statistics and Computing*, **28**(4), 869–890.

Examples

```
N <- 100
p<- 100
set.seed(100)
library(bartBMA)
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+
5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
5*xcovtest[,5]+epsilontest
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
only_max_num_trees = 1,split_rule_node = 0)
```

---

bartBMA_with_ITEs_exact_par

Prediction intervals for bart-bma output obtained using linear algebra to obtain means and variances, and using bisection to find the quantiles of the mixture of t distributions.

Description

This function produces prediction intervals for bart-bma output.

Usage

```
bartBMA_with_ITEs_exact_par(
  l_quant,
  u_quant,
  newdata = NULL,
  update_resids = 1,
  num_cores = 1,
  root_alg_precision = 1e-05,
  x_covariates,
  z_train,
  y_train,
  a = 3,
  nu = 3,
```

```
  sigquant = 0.9,
  c = 1000,
  pen = 12,
  num_cp = 20,
  x.test = matrix(0, 0, 0),
  num_rounds = 5,
  alpha = 0.95,
  beta = 2,
  split_rule_node = 0,
  gridpoint = 0,
  maxOWsize = 100,
  num_splits = 5,
  gridsize = 10,
  zero_split = 1,
  only_max_num_trees = 1,
  min_num_obs_for_split = 2,
  min_num_obs_after_split = 2,
  exact_residuals = 1,
  spike_tree = 0,
  s_t_hyperprior = 1,
  p_s_t = 0.5,
  a_s_t = 1,
  b_s_t = 3,
  lambda_poisson = 10,
  less_greedy = 0
)
```

Arguments

l_quant	Lower quantile of credible intervals for the ITEs, CATT, CATNT.
u_quant	Upper quantile of credible intervals for the ITEs, CATT, CATNT.
newdata	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
update_resids	Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residuals.
num_cores	Number of cores used in parallel.
root_alg_precision	The algorithm should obtain approximate bounds that are within the distance root_alg_precision of the true quantile for the chosen average of models.
x_covariates	Covaraite matrix for training bartBMA.
z_train	treatment vector for traiing bartBMA.
y_train	outcome vector for training bartBMA.
a	This is a parameter that influences the variance of terminal node parameter values. Default value a=3.

nu — This is a hyperparameter in the distribution of the variance of the error term. THe inverse of the variance is distributed as Gamma (nu/2, nu*lambda/2). Default value nu=3.

sigquant — Calibration quantile for the inverse chi-squared prior on the variance of the error term.

c — This determines the size of Occam's Window

pen — This is a parameter used by the Pruned Exact Linear Time Algorithm when finding changepoints. Default value pen=12.

num_cp — This is a number between 0 and 100 that determines the proportion of changepoints proposed by the changepoint detection algorithm to keep when growing trees. Default num_cp=20.

x.test — Test data covariate matrix. Default x.test=matrix(0.0,0,0).

num_rounds — Number of trees. (Maximum number of trees in a sum-of-tree model). Default num_rounds=5.

alpha — Parameter in prior probability of tree node splitting. Default alpha=0.95

beta — Parameter in prior probability of tree node splitting. Default beta=1

split_rule_node — Binary variable. If equals 1, then find a new set of potential splitting points via a changepoint algorithm after adding each split to a tree. If equals zero, use the same set of potential split points for all splits in a tree. Default split_rule_node=0.

gridpoint — Binary variable. If equals 1, then a grid search changepoint detection algorithm will be used. If equals 0, then the Pruned Exact Linear Time (PELT) changepoint detection algorithm will be used (Killick et al. 2012). Default gridpoint=0.

maxOWsize — Maximum number of models to keep in Occam's window. Default maxOWsize=100.

num_splits — Maximum number of splits in a tree

gridsize — This integer determines the size of the grid across which to search if gridpoint=1 when finding changepoints for constructing trees.

zero_split — Binary variable. If equals 1, then zero split trees can be included in a sum-of-trees model. If equals zero, then only trees with at least one split can be included in a sum-of-trees model.

only_max_num_trees — Binary variable. If equals 1, then only sum-of-trees models containing the maximum number of trees, num_rounds, are selected. If equals 0, then sum-of-trees models containing less than num_rounds trees can be selected. The default is only_max_num_trees=1.

min_num_obs_for_split — This integer determines the minimum number of observations in a (parent) tree node for the algorithm to consider potential splits of the node.

min_num_obs_after_split — This integer determines the minimum number of observations in a child node resulting from a split in order for a split to occur. If the left or right chikd node has less than this number of observations, then the split can not occur.

`exact_residuals`
: Binary variable. If equal to 1, then trees are added to sum-of-tree models within each round of the algorithm by detecting changepoints in the exact residuals. If equals zero, then changepoints are detected in residuals that are constructed from approximate predictions.

`spike_tree`
: If equal to 1, then the Spike-and-Tree prior will be used, otherwise the standard BART prior will be used. The number of splitting variables has a beta-binomial prior. The number of terminal nodes has a truncated Poisson prior, and then a uniform prior is placed on the set of valid constructions of trees given the splitting variables and number of terminal nodes.

`s_t_hyperprior`
: If equals 1 and spike_tree equals 1, then a beta distribution hyperprior is placed on the variable inclusion probabilities for the spike and tree prior. The hyperprior parameters are a_s_t and b_s_t.

`p_s_t`
: If spike_tree=1 and s_t_hyperprior=0, then p_s_t is the prior variable inclusion probability.

`a_s_t`
: If spike_tree=1 and s_t_hyperprior=1, then a_s_t is a parameter of a beta distribution hyperprior.

`b_s_t`
: If spike_tree=1 and s_t_hyperprior=1, then b_s_t is a parameter of a beta distribution hyperprior.

`lambda_poisson`
: This is a parameter for the Spike-and-Tree prior. It is the parameter for the (truncated and conditional on the number of splitting variables) Poisson prior on the number of terminal nodes.

`less_greedy`
: If equal to one, then a less greedy model search algorithm is used.

Value

The output is a list of length 4:

`ITE_intervals`
: A 3 by n matrix, where n is the number of observations. The first row gives the l_quant*100 quantiles of the individual treatment effects. The second row gives the medians of the ITEs. The third row gives the u_quant*100 quantiles of the ITEs.

`ITE_estimates`
: An n by 1 matrix containing the Individual Treatment Effect estimates.

`CATE_estimate`
: The Conditional Average Treatment Effect Estimates

`CATE_Interval`
: A 3 by 1 matrix. The first element is the l_quant*100 quantile of the CATE distribution, the second element is the median of the CATE distribution, and the thied element is the u_quant*100 quantile of the CATE distribution.

Examples

```
## Not run:
#Example of BART-BMA for ITE estimation
#Applied to data simulations from Hahn et al. (2020, Bayesian Analysis)
#"Bayesian Regression Tree Models for Causal Inference: Regularization, Confounding,
# and Heterogeneous Effects
n <- 250
x1 <- rnorm(n)
```

```
x2 <- rnorm(n)
x3 <- rnorm(n)
x4 <- rbinom(n,1,0.5)
x5 <- as.factor(sample( LETTERS[1:3], n, replace=TRUE))

p= 0
xnoise = matrix(rnorm(n*p), nrow=n)
x5A <- ifelse(x5== 'A',1,0)
x5B <- ifelse(x5== 'B',1,0)
x5C <- ifelse(x5== 'C',1,0)

x_covs_train <- cbind(x1,x2,x3,x4,x5A,x5B,x5C,xnoise)

#Treatment effect
#tautrain <- 3
tautrain <- 1+2*x_covs_train[,2]*x_covs_train[,4]

#Prognostic function
mutrain <- 1 + 2*x_covs_train[,5] -1*x_covs_train[,6]-4*x_covs_train[,7] +
x_covs_train[,1]*x_covs_train[,3]
sd_mtrain <- sd(mutrain)
utrain <- runif(n)
#pitrain <- 0.8*pnorm((3*mutrain/sd_mtrain)-0.5*x_covs_train[,1])+0.05+utrain/10
pitrain <- 0.5
ztrain <- rbinom(n,1,pitrain)
ytrain <- mutrain + tautrain*ztrain
#pihattrain <- pbart(x_covs_train,ztrain )$prob.train.mean

#set lower and upper quantiles for intervals
lbound <- 0.025
ubound <- 0.975

example_output <- bartBMA_with_ITEs_exact_par(l_quant = lbound,
                                              u_quant= ubound,
                                              x_covariates = x_covs_train,
                                              z_train = ztrain,
                                              y_train = ytrain)

## End(Not run)
```

`ITEs_bartBMA` *ITE Predictions (in-sample) using bartBMA and the method described by Hill (2011)*

Description

This function produces ITE Predictions (in-sample) using bartBMA and the method described by Hill (2011).

Usage

```
ITEs_bartBMA(
  x_covariates,
  z_train,
  y_train,
  a = 3,
  nu = 3,
  sigquant = 0.9,
  c = 1000,
  pen = 12,
  num_cp = 20,
  x.test = matrix(0, 0, 0),
  num_rounds = 5,
  alpha = 0.95,
  beta = 2,
  split_rule_node = 0,
  gridpoint = 0,
  maxOWsize = 100,
  num_splits = 5,
  gridsize = 10,
  zero_split = 1,
  only_max_num_trees = 1,
  min_num_obs_for_split = 2,
  min_num_obs_after_split = 2
)
```

Arguments

x_covariates	Covaraite matrix for training bartBMA.
z_train	treatment vector for traiing bartBMA.
y_train	outcome vector for training bartBMA.
a	This is a parameter that influences the variance of terminal node parameter values. Default value a=3.
nu	This is a hyperparameter in the distribution of the variance of the error term. THe inverse of the variance is distributed as Gamma (nu/2, nu*lambda/2). Default value nu=3.
sigquant	Calibration quantile for the inverse chi-squared prior on the variance of the error term.
c	This determines the size of Occam's Window
pen	This is a parameter used by the Pruned Exact Linear Time Algorithm when finding changepoints. Default value pen=12.
num_cp	This is a number between 0 and 100 that determines the proportion of changepoints proposed by the changepoint detection algorithm to keep when growing trees. Default num_cp=20.
x.test	Test data covariate matrix. Default x.test=matrix(0.0,0,0).

num_rounds Number of trees. (Maximum number of trees in a sum-of-tree model). Default num_rounds=5.

alpha Parameter in prior probability of tree node splitting. Default alpha=0.95

beta Parameter in prior probability of tree node splitting. Default beta=1

split_rule_node
Binary variable. If equals 1, then find a new set of potential splitting points via a changepoint algorithm after adding each split to a tree. If equals zero, use the same set of potential split points for all splits in a tree. Default split_rule_node=0.

gridpoint Binary variable. If equals 1, then a grid search changepoint detection algorithm will be used. If equals 0, then the Pruned Exact Linear Time (PELT) changepoint detection algorithm will be used (Killick et al. 2012). Default gridpoint=0.

maxOWsize Maximum number of models to keep in Occam's window. Default maxOWsize=100.

num_splits Maximum number of splits in a tree

gridsize This integer determines the size of the grid across which to search if gridpoint=1 when finding changepoints for constructing trees.

zero_split Binary variable. If equals 1, then zero split trees can be included in a sum-of-trees model. If equals zero, then only trees with at least one split can be included in a sum-of-trees model.

only_max_num_trees
Binary variable. If equals 1, then only sum-of-trees models containing the maximum number of trees, num_rounds, are selected. If equals 0, then sum-of-trees models containing less than num_rounds trees can be selected. The default is only_max_num_trees=1.

min_num_obs_for_split
This integer determines the minimum number of observations in a (parent) tree node for the algorithm to consider potential splits of the node.

min_num_obs_after_split
This integer determines the minimum number of observations in a child node resulting from a split in order for a split to occur. If the left or right chikd node has less than this number of observations, then the split can not occur.

Value

A list of length 2. The first element is A vector of Individual Treatment Effect Estimates. The second element is a bartBMA object (i.e. the trained BART-BMA model).

Examples

```
n <- 250
x1 <- rnorm(n)
x2 <- rnorm(n)
x3 <- rnorm(n)
x4 <- rbinom(n,1,0.5)
x5 <- as.factor(sample( LETTERS[1:3], n, replace=TRUE))

p= 0
```

```
xnoise = matrix(rnorm(n*p), nrow=n)
x5A <- ifelse(x5== 'A',1,0)
x5B <- ifelse(x5== 'B',1,0)
x5C <- ifelse(x5== 'C',1,0)

x_covs_train <- cbind(x1,x2,x3,x4,x5A,x5B,x5C,xnoise)

#Treatment effect
#tautrain <- 3
tautrain <- 1+2*x_covs_train[,2]*x_covs_train[,4]

#Prognostic function
mutrain <- 1 + 2*x_covs_train[,5] -1*x_covs_train[,6]-4*x_covs_train[,7] +
x_covs_train[,1]*x_covs_train[,3]
sd_mtrain <- sd(mutrain)
utrain <- runif(n)
#pitrain <- 0.8*pnorm((3*mutrain/sd_mtrain)-0.5*x_covs_train[,1])+0.05+utrain/10
pitrain <- 0.5
ztrain <- rbinom(n,1,pitrain)
ytrain <- mutrain + tautrain*ztrain
#pihattrain <- pbart(x_covs_train,ztrain )$prob.train.mean

#set lower and upper quantiles for intervals
lbound <- 0.025
ubound <- 0.975

example_output <- ITEs_bartBMA(x_covariates = x_covs_train,
                               z_train = ztrain,
                               y_train = ytrain)
```

`ITEs_bartBMA_exact_par` *Estimate ITEs and obtain credible intervals (in-sample or out-of-sample).*

Description

This function takes a set of sum of tree models obtained from ITEs_bartBMA, and then estimates ITEs, and obtains prediction intervals.

Usage

```
ITEs_bartBMA_exact_par(
  object,
  l_quant,
  u_quant,
  newdata = NULL,
  update_resids = 1,
  num_cores = 1,
```

```
  root_alg_precision = 1e-05,
  training_data
)
```

Arguments

object	Output from ITEs_bartBMA of class ITE_ests.bartBMA.
l_quant	Lower quantile of credible intervals for the ITEs, CATT, CATNT.
u_quant	Upper quantile of credible intervals for the ITEs, CATT, CATNT.
newdata	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
update_resids	Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residuals.
num_cores	Number of cores used in parallel.
root_alg_precision	The algorithm should obtain approximate bounds that are within the distance root_alg_precision of the true quantile for the chosen average of models.
training_data	The training data matrix

Value

The output is a list of length 4:

ITE_intervals	A 3 by n matrix, where n is the number of observations. The first row gives the l_quant*100 quantiles of the individual treatment effects. The second row gives the medians of the ITEs. The third row gives the u_quant*100 quantiles of the ITEs.
ITE_estimates	An n by 1 matrix containing the Individual Treatment Effect estimates.
CATE_estimate	The Conditional Average Treatment Effect Estimates
CATE_Interval	A 3 by 1 matrix. The first element is the l_quant*100 quantile of the CATE distribution, the second element is the median of the CATE distribution, and the thied element is the u_quant*100 quantile of the CATE distribution.

Examples

```
## Not run:
#Example of BART-BMA for ITE estimation
# Applied to data simulations from Hahn et al. (2020, Bayesian Analysis)
# "Bayesian Regression Tree Models for Causal Inference: Regularization,
# Confounding, and Heterogeneous Effects
n <- 250
x1 <- rnorm(n)
x2 <- rnorm(n)
x3 <- rnorm(n)
```

```
x4 <- rbinom(n,1,0.5)
x5 <- as.factor(sample( LETTERS[1:3], n, replace=TRUE))

p= 0
xnoise = matrix(rnorm(n*p), nrow=n)
x5A <- ifelse(x5== 'A',1,0)
x5B <- ifelse(x5== 'B',1,0)
x5C <- ifelse(x5== 'C',1,0)

x_covs_train <- cbind(x1,x2,x3,x4,x5A,x5B,x5C,xnoise)

#Treatment effect
#tautrain <- 3
tautrain <- 1+2*x_covs_train[,2]*x_covs_train[,4]

#Prognostic function
mutrain <- 1 + 2*x_covs_train[,5] -1*x_covs_train[,6]-4*x_covs_train[,7] +
x_covs_train[,1]*x_covs_train[,3]
sd_mtrain <- sd(mutrain)
utrain <- runif(n)
#pitrain <- 0.8*pnorm((3*mutrain/sd_mtrain)-0.5*x_covs_train[,1])+0.05+utrain/10
pitrain <- 0.5
ztrain <- rbinom(n,1,pitrain)
ytrain <- mutrain + tautrain*ztrain
#pihattrain <- pbart(x_covs_train,ztrain )$prob.train.mean

#set lower and upper quantiles for intervals
lbound <- 0.025
ubound <- 0.975


trained_bbma <- ITEs_bartBMA(x_covariates = x_covs_train,
                             z_train = ztrain,
                             y_train = ytrain)

example_output <- ITEs_bartBMA_exact_par(trained_bbma[[2]],
                                         l_quant = lbound,
                                         u_quant= ubound,
                                         training_data = x_covs_train)

## End(Not run)
```

`ITEs_CATT_bartBMA_exact_par` *Estimate ITEs, CATE, CATT, CATNT and obtain credible intervals (in-sample or out-of-sample).*

Description

This function takes a set of sum of tree models obtained from ITEs_bartBMA, and then estimates ITEs, and the CATE, CATT, and CATNT and obtains prediction intervals

Usage

```
ITEs_CATT_bartBMA_exact_par(
  object,
  l_quant,
  u_quant,
  newdata = NULL,
  update_resids = 1,
  num_cores = 1,
  root_alg_precision = 1e-05,
  training_data,
  zvec
)
```

Arguments

object	Output from ITEs_bartBMA of class ITE_ests.bartBMA.
l_quant	Lower quantile of credible intervals for the ITEs, CATT, CATNT.
u_quant	Upper quantile of credible intervals for the ITEs, CATT, CATNT.
newdata	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
update_resids	Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residuals.
num_cores	Number of cores used in parallel.
root_alg_precision	The algorithm should obtain approximate bounds that are within the distance root_alg_precision of the true quantile for the chosen average of models.
training_data	The training data matrix
zvec	The treatment indicator vector. Training data treatment vector for insample predictions, test data treatment vector for out of sample predictions.

Value

The output is a list of length 8:

ITE_intervals	A 3 by n matrix, where n is the number of observations. The first row gives the l_quant*100 quantiles of the individual treatment effects. The second row gives the medians of the ITEs. The third row gives the u_quant*100 quantiles of the ITEs.
ITE_estimates	An n by 1 matrix containing the Individual Treatment Effect estimates.
CATE_estimate	The Conditional Average Treatment Effect Estimate
CATE_Interval	A 3 by 1 matrix. The first element is the l_quant*100 quantile of the CATE distribution, the second element is the median of the CATE distribution, and the thied element is the u_quant*100 quantile of the CATE distribution.

CATT_estimate	The Conditional Average Treatment Effect on the Treated Estimate
CATT_Interval	A 3 by 1 matrix. The first element is the l_quant*100 quantile of the CATT distribution, the second element is the median of the CATT distribution, and the thied element is the u_quant*100 quantile of the CATT distribution.
CATNT_estimate	The Conditional Average Treatment Effect on the Not Treated Estimate
CATNT_Interval	A 3 by 1 matrix. The first element is the l_quant*100 quantile of the CATNT distribution, the second element is the median of the CATNT distribution, and the thied element is the u_quant*100 quantile of the CATNT distribution.

Examples

```
## Not run:
#Example of BART-BMA for ITE estimation
# Applied to data simulations from Hahn et al. (2020, Bayesian Analysis)
# "Bayesian Regression Tree Models for Causal Inference: Regularization,
# Confounding, and Heterogeneous Effects
n <- 250
x1 <- rnorm(n)
x2 <- rnorm(n)
x3 <- rnorm(n)
x4 <- rbinom(n,1,0.5)
x5 <- as.factor(sample( LETTERS[1:3], n, replace=TRUE))

p= 0
xnoise = matrix(rnorm(n*p), nrow=n)
x5A <- ifelse(x5== 'A',1,0)
x5B <- ifelse(x5== 'B',1,0)
x5C <- ifelse(x5== 'C',1,0)

x_covs_train <- cbind(x1,x2,x3,x4,x5A,x5B,x5C,xnoise)

#Treatment effect
#tautrain <- 3
tautrain <- 1+2*x_covs_train[,2]*x_covs_train[,4]

#Prognostic function
mutrain <- 1 + 2*x_covs_train[,5] -1*x_covs_train[,6]-4*x_covs_train[,7] +
x_covs_train[,1]*x_covs_train[,3]
sd_mtrain <- sd(mutrain)
utrain <- runif(n)
#pitrain <- 0.8*pnorm((3*mutrain/sd_mtrain)-0.5*x_covs_train[,1])+0.05+utrain/10
pitrain <- 0.5
ztrain <- rbinom(n,1,pitrain)
ytrain <- mutrain + tautrain*ztrain
#pihattrain <- pbart(x_covs_train,ztrain )$prob.train.mean

#set lower and upper quantiles for intervals
lbound <- 0.025
ubound <- 0.975
```

```
trained_bbma <- ITEs_bartBMA(x_covariates = x_covs_train,
                             z_train = ztrain,
                             y_train = ytrain)

example_output <- ITEs_CATT_bartBMA_exact_par(trained_bbma[[2]],
                                              l_quant = lbound,
                                              u_quant= ubound,
                                              training_data = x_covs_train,
                                              zvec = ztrain,
                                              num_cores = 1)

## End(Not run)
```

`predict_bartBMA` *Predictions for a new dataset using an existing bartbma object*

Description

This function produces predictions for a new dataset using a previously obtained bartBMA object.

Usage

```
predict_bartBMA(object, newdata)
```

Arguments

`object`	A bartBMA object obtained using the barBMA function.
`newdata`	Covariate matrix for new dataset.

Value

A vector of predictions for the new dataset.

Examples

```
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
```

```
                              only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
predict_bartBMA(bart_bma_example,newdata=xcovtest)
```

`predict_probit_bartBMA` *Predictions for a new dataset using an existing probit_bartBMA object*

Description

This function produces predictions for a new dataset using a previously obtained bartBMA object.

Usage

```
predict_probit_bartBMA(object, newdata)
```

Arguments

`object`	A probit_bartBMA object obtained using the probit_bartBMA function.
`newdata`	Covariate matrix for new dataset.

Value

The output is a list of length 2:

`probs`	A vector of estimated probabilities for newdata.
`pred_binary`	A vector of binary predictions for newdata.

Examples

```
#Example from BART package (McCulloch et al. 2019)
set.seed(99)
n=100
x = sort(-2+4*runif(n))
X=matrix(x,ncol=1)
f = function(x) {return((1/2)*x^3)}
FL = function(x) {return(exp(x)/(1+exp(x)))}
pv = FL(f(x))
y = rbinom(n,1,pv)

trained_probit_bbma <- probit_bartBMA(x.train = X,y.train = y)

np=100
xp=-2+4*(1:np)/np
Xp=matrix(xp,ncol=1)

predict_probit_bartBMA(trained_probit_bbma,Xp)
```

`preds_bbma_lin_alg` *Predictions for bart-bma output obtained from the posterior probability weighted averaged of the posterior means for each model*

Description

This function produces predictions from BART-BMA by obtaining the posterior probability weighted averaged of the posterior means for each model.

Usage

```
preds_bbma_lin_alg(
  object,
  num_iter,
  burnin,
  newdata = NULL,
  update_resids = 1,
  trainingdata
)
```

Arguments

`object`	bartBMA object obtained from function bartBMA
`num_iter`	Total number of iterations of the Gibbs sampler (including burn-in).
`burnin`	Number of burn-on iterations of the Gibbs sampler.
`newdata`	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
`update_resids`	Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residua;s.
`trainingdata`	The matrix of training data.

Value

A vector of predictions.

Examples

```
#set the seed
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
```

```
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
preds_bbma_lin_alg(bart_bma_example,1000,100,newdata=xcovtest)
```

`pred_expectation_intervals_bbma_GS` *Prediction intervals for bart-bma output*

Description

This function produces prediction intervals for f(x) in BART-BMA by post-hoc Gibbs-sampling from the full conditionals of the terminal node parameters and the variance of the error term. See Hernandez et al. (2018) Appendix D for details.

Usage

```
pred_expectation_intervals_bbma_GS(
  object,
  num_iter,
  burnin,
  l_quant,
  u_quant,
  newdata = NULL,
  update_resids = 1
)
```

Arguments

`object`	bartBMA object obtained from function bartBMA
`num_iter`	Total number of iterations of the Gibbs sampler (including burn-in).
`burnin`	Number of burn-on iterations of the Gibbs sampler.
`l_quant`	Lower quartile of the prediction interval.
`u_quant`	Upper quartile of the prediction interval.
`newdata`	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.

update_resids Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residua;s.

Value

The output is a list of length 2:

PI A 3 by n matrix, where n is the number of observations. The first row gives the l_quant*100 quantiles of f(x). The second row gives the medians of f(x). The third row gives the u_quant*100 quantiles of f(x).

meanpreds An n by 1 matrix containing the estimated means of f(x).

Examples

```
#load the package
library(bartBMA)
#set the seed
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
pred_expectation_intervals_bbma_GS(bart_bma_example,1000,100,0.025,0.975,
newdata=NULL,update_resids=1)
```

---

pred_intervals_bbma_GS

Prediction intervals for bart-bma output

Description

This function produces prediction intervals for BART-BMA estimates by post-hoc Gibbs-sampling from the full conditionals of the terminal node parameters and the variance of the error term. See Hernandez et al. (2018) Appendix D for details.

Usage

```
pred_intervals_bbma_GS(
  object,
  num_iter,
  burnin,
  l_quant,
  u_quant,
  newdata = NULL,
  update_resids = 1
)
```

Arguments

object	bartBMA object obtained from function bartBMA
num_iter	Total number of iterations of the Gibbs sampler (including burn-in).
burnin	Number of burn-on iterations of the Gibbs sampler.
l_quant	Lower quartile of the prediction interval.
u_quant	Upper quartile of the prediction interval.
newdata	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
update_resids	Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residua;s.

Value

The output is a list of length 2:

PI	A 3 by n matrix, where n is the number of observations. The first row gives the l_quant*100 quantiles. The second row gives the medians. The third row gives the u_quant*100 quantiles.
meanpreds	An n by 1 matrix containing the estimated means.

Examples

```
#load the package
library(bartBMA)
#set the seed
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
```

```
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
pred_intervals_bbma_GS(bart_bma_example,1000,100,0.025,0.975,newdata=NULL,update_resids=1)
```

`pred_intervals_new_initials_GS` *Prediction intervals for bart-bma output*

Description

This function produces prediction intervals for BART-BMA estimates by post-hoc Gibbs-sampling from the full conditionals of the terminal node parameters and the variance of the error term. See Hernandez et al. (2018) Appendix D for details.

Usage

```
pred_intervals_new_initials_GS(
  object,
  num_iter,
  burnin,
  l_quant,
  u_quant,
  newdata = NULL,
  update_resids = 1,
  trainingdata
)
```

Arguments

`object`	bartBMA object obtained from function bartBMA
`num_iter`	Total number of iterations of the Gibbs sampler (including burn-in).
`burnin`	Number of burn-on iterations of the Gibbs sampler.
`l_quant`	Lower quartile of the prediction interval.
`u_quant`	Upper quartile of the prediction interval.
`newdata`	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.

update_resids Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residuals.

trainingdata The matrix of training data.

Value

The output is a list of length 2:

PI A 3 by n matrix, where n is the number of observations. The first row gives the l_quant*100 quantiles. The second row gives the medians. The third row gives the u_quant*100 quantiles.

meanpreds An n by 1 matrix containing the estimated means.

Examples

```
#load the package
library(bartBMA)
#set the seed
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
pred_intervals_new_initials_GS(bart_bma_example,1000,100,0.025,0.975,
newdata=NULL,update_resids=1,xcov)
```

---

pred_ints_exact *Prediction intervals for bart-bma output obtained using linear algebra to obtain means and variances, and using bisection to find the quantiles of the mixture of t distributions.*

Description

This function produces prediction intervals for bart-bma output.

Usage

```
pred_ints_exact(
  object,
  l_quant,
  u_quant,
  newdata = NULL,
  num_cores = 1,
  root_alg_precision = 1e-05
)
```

Arguments

object	bartBMA object obtained from function bartBMA
l_quant	Lower quantile of credible intervals for the ITEs, CATT, CATNT.
u_quant	Upper quantile of credible intervals for the ITEs, CATT, CATNT.
newdata	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
num_cores	Number of cores used in parallel.
root_alg_precision	The algorithm should obtain approximate bounds that are within the distance root_alg_precision of the true quantile for the chosen average of models.

Value

The output is a list of length 2:

PI	A 3 by n matrix, where n is the number of observations. The first row gives the l_quant*100 quantiles. The second row gives the medians. The third row gives the u_quant*100 quantiles.
meanpreds	An n by 1 matrix containing the estimated means.

Examples

```
#load the package
library(bartBMA)
#set the seed
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
```

```
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
pred_ints_exact(bart_bma_example,0.025,0.975,newdata=NULL,num_cores=1)
```

`pred_ints_exact_par` *Prediction intervals for bart-bma output obtained using linear algebra to obtain means and variances, and using bisection to find the quantiles of the mixture of t distributions.*

Description

This function produces prediction intervals for bart-bma output.

Usage

```
pred_ints_exact_par(
  object,
  l_quant,
  u_quant,
  newdata = NULL,
  num_cores = 1,
  root_alg_precision = 1e-05
)
```

Arguments

`object`	bartBMA object obtained from function bartBMA
`l_quant`	Lower quantile of credible intervals for the ITEs, CATT, CATNT.
`u_quant`	Upper quantile of credible intervals for the ITEs, CATT, CATNT.
`newdata`	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
`num_cores`	Number of cores used in parallel.
`root_alg_precision`	The algorithm should obtain approximate bounds that are within the distance root_alg_precision of the true quantile for the chosen average of models.

Value

The output is a list of length 2:

PI	A 3 by n matrix, where n is the number of observations. The first row gives the l_quant*100 quantiles. The second row gives the medians. The third row gives the u_quant*100 quantiles.
meanpreds	An n by 1 matrix containing the estimated means.

Examples

```
## Not run:
#load the package
library(bartBMA)
#set the seed
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
pred_ints_exact_par(bart_bma_example,0.025,0.975,newdata=NULL,num_cores=1)

## End(Not run)
```

---

pred_means_bbma_GS *Predictions for bart-bma output obtained from a Gibbs sampler*

Description

This function produces predictions from BART-BMA by post-hoc Gibbs-sampling from the full conditionals of the terminal node parameters and the variance of the error term. See Hernandez et al. (2018) Appendix D for details.

Usage

```
pred_means_bbma_GS(object, num_iter, burnin, newdata = NULL, update_resids = 1)
```

Arguments

object	bartBMA object obtained from function bartBMA
num_iter	Total number of iterations of the Gibbs sampler (including burn-in).
burnin	Number of burn-on iterations of the Gibbs sampler.
newdata	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
update_resids	Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residua;s.

Value

The output is a vector of predictions.

Examples

```
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
pred_means_bbma_GS(bart_bma_example,1000,100,newdata=NULL,update_resids=1)
```

---

pred_means_bbma_new_initials_GS

Predictions for bart-bma output obtained from a Gibbs sampler

---

Description

This function produces predictions from BART-BMA by post-hoc Gibbs-sampling from the full conditionals of the terminal node parameters and the variance of the error term. See Hernandez et al. (2018) Appendix D for details.

Usage

```
pred_means_bbma_new_initials_GS(
  object,
  num_iter,
  burnin,
  newdata = NULL,
  update_resids = 1,
  trainingdata
)
```

Arguments

object	bartBMA object obtained from function bartBMA
num_iter	Total number of iterations of the Gibbs sampler (including burn-in).
burnin	Number of burn-on iterations of the Gibbs sampler.
newdata	Test data for which predictions are to be produced. Default = NULL. If NULL, then produces prediction intervals for training data if no test data was used in producing the bartBMA object, or produces prediction intervals for the original test data if test data was used in producing the bartBMA object.
update_resids	Option for whether to update the partial residuals in the gibbs sampler. If equal to 1, updates partial residuals, if equal to zero, does not update partial residuals. The defaullt setting is to update the partial residua;s.
trainingdata	The matrix of training data.

Value

The output is a vector of predictions.

Examples

```
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the prediction intervals
pred_means_bbma_new_initials_GS(bart_bma_example,1000,100,newdata=NULL,update_resids=1,xcovtest)
```

probit_bartBMA *Probit BART_BMA for classification of a binary variable*

Description

This is an implementation of Bayesian Additive Regression Trees (Chipman et al. 2018) using Bayesian Model Averaging (Hernandez et al. 2018).

Usage

```
probit_bartBMA(x.train, ...)

## Default S3 method:
probit_bartBMA(
  x.train,
  y.train,
  a = 3,
  nu = 3,
  sigquant = 0.9,
  c = 1000,
  pen = 12,
  num_cp = 20,
  x.test = matrix(0, 0, 0),
  num_rounds = 5,
  alpha = 0.95,
  beta = 2,
  split_rule_node = 0,
  gridpoint = 0,
  maxOWsize = 100,
  num_splits = 5,
  gridsize = 10,
  zero_split = 1,
  only_max_num_trees = 1,
  min_num_obs_for_split = 2,
  min_num_obs_after_split = 2,
  exact_residuals = 1,
  spike_tree = 0,
  s_t_hyperprior = 1,
  p_s_t = 0.5,
  a_s_t = 1,
  b_s_t = 3,
  lambda_poisson = 10,
  less_greedy = 0,
  ...
)
```

Arguments

x.train	Training data covariate matrix
...	Further arguments.
y.train	Training data outcome vector.
a	This is a parameter that influences the variance of terminal node parameter values. Default value a=3.
nu	This is a hyperparameter in the distribution of the variance of the error term. THe inverse of the variance is distributed as Gamma (nu/2, nu*lambda/2). Default value nu=3.
sigquant	Calibration quantile for the inverse chi-squared prior on the variance of the error term.
c	This determines the size of Occam's Window
pen	This is a parameter used by the Pruned Exact Linear Time Algorithm when finding changepoints. Default value pen=12.
num_cp	This is a number between 0 and 100 that determines the proportion of changepoints proposed by the changepoint detection algorithm to keep when growing trees. Default num_cp=20.
x.test	Test data covariate matrix. Default x.test=matrix(0.0,0,0).
num_rounds	Number of trees. (Maximum number of trees in a sum-of-tree model). Default num_rounds=5.
alpha	Parameter in prior probability of tree node splitting. Default alpha=0.95
beta	Parameter in prior probability of tree node splitting. Default beta=1
split_rule_node	Binary variable. If equals 1, then find a new set of potential splitting points via a changepoint algorithm after adding each split to a tree. If equals zero, use the same set of potential split points for all splits in a tree. Default split_rule_node=0.
gridpoint	Binary variable. If equals 1, then a grid search changepoint detection algorithm will be used. If equals 0, then the Pruned Exact Linear Time (PELT) changepoint detection algorithm will be used (Killick et al. 2012). Default gridpoint=0.
maxOWsize	Maximum number of models to keep in Occam's window. Default maxOWsize=100.
num_splits	Maximum number of splits in a tree
gridsize	This integer determines the size of the grid across which to search if gridpoint=1 when finding changepoints for constructing trees.
zero_split	Binary variable. If equals 1, then zero split trees can be included in a sum-of-trees model. If equals zero, then only trees with at least one split can be included in a sum-of-trees model.
only_max_num_trees	Binary variable. If equals 1, then only sum-of-trees models containing the maximum number of trees, num_rounds, are selected. If equals 0, then sum-of-trees models containing less than num_rounds trees can be selected. The default is only_max_num_trees=1.

`min_num_obs_for_split`
: This integer determines the minimum number of observations in a (parent) tree node for the algorithm to consider potential splits of the node.

`min_num_obs_after_split`
: This integer determines the minimum number of observations in a child node resulting from a split in order for a split to occur. If the left or right chikd node has less than this number of observations, then the split can not occur.

`exact_residuals`
: Binary variable. If equal to 1, then trees are added to sum-of-tree models within each round of the algorithm by detecting changepoints in the exact residuals. If equals zero, then changepoints are detected in residuals that are constructed from approximate predictions.

`spike_tree`
: If equal to 1, then the Spike-and-Tree prior will be used, otherwise the standard BART prior will be used. The number of splitting variables has a beta-binomial prior. The number of terminal nodes has a truncated Poisson prior, and then a uniform prior is placed on the set of valid constructions of trees given the splitting variables and number of terminal nodes.

`s_t_hyperprior`
: If equals 1 and spike_tree equals 1, then a beta distribution hyperprior is placed on the variable inclusion probabilities for the spike and tree prior. The hyperprior parameters are a_s_t and b_s_t.

`p_s_t`
: If spike_tree=1 and s_t_hyperprior=0, then p_s_t is the prior variable inclusion probability.

`a_s_t`
: If spike_tree=1 and s_t_hyperprior=1, then a_s_t is a parameter of a beta distribution hyperprior

`b_s_t`
: If spike_tree=1 and s_t_hyperprior=1, then b_s_t is a parameter of a beta distribution hyperprior

`lambda_poisson`
: This is a parameter for the Spike-and-Tree prior. It is the parameter for the (truncated and conditional on the number of splitting variables) Poisson prior on the number of terminal nodes.

`less_greedy`
: If equal to one, then a less greedy model search algorithm is used.

Value

The following objects are returned by bartbma:

`fitted.values`
: The vector of predictions of the outcome for all training observations.

`sumoftrees`
: This is a list of lists of matrices. The outer list corresponds to a list of sum-of-tree models, and each element of the outer list is a list of matrices describing the structure of the trees within a sum-of-tree model. See details.

`obs_to_termNodesMatrix`
: This is a list of lists of matrices. The outer list corresponds to a list of sum-of-tree models, and each element of the outer list is a list of matrices describing to which node each of the observations is allocated to at all depths of each tree within a sum-of-tree model. See details.

`bic`
: This is a vector of BICs for each sum-of-tree model.

test.preds	A vector of test data predictions. This output only is given if there is test data in the input.
sum_residuals	CURRENTLY INCORRECT OUTPUT. A List (over sum-of-tree models) of lists (over single trees in a model) of vectors of partial residuals. Unless the maximum number of trees in a model is one, in which case the output is a list (over single tree models) of vectors of partial residuals, which are all equal to the outcome vector.
numvars	This is the total number of variables in the input training data matrix.
call	match.call returns a call in which all of the specified arguments are specified by their full names.
y_minmax	Range of the input training data outcome vector.
response	Input taining data outcome vector.
nrowTrain	number of observations in the input training data.
sigma	sd(y.train)/(max(y.train)-min(y.train))
a	input parameter
nu	input parameter
lambda	parameter determined by the inputs sigma, sigquant, and nu
fitted.probs	In-sample fitted probabilities
fitted.classes	In-sample fitted classes

Examples

```
#Example from BART package (McCulloch et al. 2019)
set.seed(99)
n=100
x = sort(-2+4*runif(n))
X=matrix(x,ncol=1)
f = function(x) {return((1/2)*x^3)}
FL = function(x) {return(exp(x)/(1+exp(x)))}
pv = FL(f(x))
y = rbinom(n,1,pv)
probit_bartBMA(x.train = X,y.train = y)
```

---

varImpScores *Variable importances as defined by Hernandez et al. (2018)*

Description

This measure defines the importance of a variable as the model-probability weighted sum of the number of splits on the variable of interest, divided by the sum over all variables of such weighted counts of splits.

Usage

```
varImpScores(object)
```

Arguments

object	A bartBMA object obtained using the barBMA function.

Value

A vector of variable importances. The variables are ordered in the same order that they occur in columns of the input covariate matrix used to obtain the input bartBMA object.

Examples

```
#set the seed
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the variable importances
varImpScores(bart_bma_example)
```

---

varIncProb *Variable inclusion probabilities as defined by Linero (2018)*

Description

This measure defines the posterior inclusion probability of a variable as the model-probability weighted sum of indicator variables for whether the variable was used in any splitting rules in any of the trees in the sum-of-tree model.

Usage

```
varIncProb(object)
```

Arguments

object	A bartBMA object obtained using the barBMA function.

Value

A vector of posterior inclusion probabilities. The variables are ordered in the same order that they occur in columns of the input covariate matrix used to obtain the input bartBMA object.

Examples

```
#set the seed
set.seed(100)
#simulate some data
N <- 100
p<- 100
epsilon <- rnorm(N)
xcov <- matrix(runif(N*p), nrow=N)
y <- sin(pi*xcov[,1]*xcov[,2]) + 20*(xcov[,3]-0.5)^2+10*xcov[,4]+5*xcov[,5]+epsilon
epsilontest <- rnorm(N)
xcovtest <- matrix(runif(N*p), nrow=N)
ytest <- sin(pi*xcovtest[,1]*xcovtest[,2]) + 20*(xcovtest[,3]-0.5)^2+10*xcovtest[,4]+
  5*xcovtest[,5]+epsilontest

#Train the object
bart_bma_example <- bartBMA(x.train = xcov,y.train=y,x.test=xcovtest,zero_split = 1,
                            only_max_num_trees = 1,split_rule_node = 0)
#Obtain the variable importances
varIncProb(bart_bma_example)
```

Index