Package: ridgeBART (via r-universe)

October 7, 2026

Type Package

Title Bayesian Additive Regression Trees with Ridge Function Outputs

Version 1.0.2

Date 2026-05-19

Description Implements an extension of Bayesian Additive Regression Trees (BART) in which each regression tree outputs a linear combination of random ridge functions (i.e., a composition of a non-linear function like cosine, hyperbolic tangent, the rectified linear unit with an affine transformation) instead of a constant. Can be used to perform ``targeted smoothing'' in which trees split on certain covariates but output smooth functions in other covariates. For more information, see Yee, Ghosh, and Deshpande (2026+) <doi:10.48550/arXiv.2411.07984>.

URL https://github.com/ryanyee3/ridgeBART

License GPL (>= 3)

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp

NeedsCompilation yes

Author Ryan Yee [aut] (ORCID: <https://orcid.org/0009-0005-5691-4009>), Sameer K. Deshpande [aut, cre] (ORCID: <https://orcid.org/0000-0003-4116-5533>)

Maintainer Sameer K. Deshpande <sameer.deshpande@wisc.edu>

Repository https://cran.r-universe.dev

Date/Publication 2026-05-27 10:30:02 UTC

RemoteUrl https://github.com/cran/ridgeBART

RemoteRef HEAD

RemoteSha 54396aab64a09ba65ca1daa372a577abde8242d9

Contents

predict_ridgeBART 2
probit_ridgeBART 3
ridgeBART 6

Index **11**

`predict_ridgeBART` *Predicting new observations with previously fit ridgeBART models.*

Description

`predict_ridgeBART` can take the output of `ridgeBART` or `probit_ridgeBART` and use it to make predictions at new inputs.

Usage

```
predict_ridgeBART(
  fit,
  X_cont = matrix(0, nrow = 1, ncol = 1),
  X_cat = matrix(0, nrow = 1, ncol = 1),
  Z_mat = matrix(0, nrow = 1, ncol = 1),
  verbose = FALSE, print_every = 100
)
```

Arguments

`fit`	Object returned by `ridgeBART` or `probit_ridgeBART`
`X_cont`	Matrix of continuous predictors. Note all predictors must be re-scaled to lie in the interval [-1,1]. Default is a 1x1 matrix, which signals that no continuos predictors are available.
`X_cat`	Integer matrix of categorical predictors for training data. Note categorical levels should be 0-indexed. That is, if a categorical predictor has 10 levels, the values should run from 0 to 9. Default is a 1x1 matrix, which signals that no categorical predictors.
`Z_mat`	Matrix of smoothing predictors. Note, predictors must be re-scaled to lie in the interval [-1,1]. Default is a 1x1 matrix, which signals that there are no continuous predictors in the training data.
`verbose`	Logical, indicating whether or not to print message predictions are being made. Default is `FALSE`.
`print_every`	As the function loops over the MCMC samples, a message is printed to the console every `print_every` iterations. Default is 100.

Value

A matrix containing posterior samples of the regression function evaluated at the supplied inputs. The rows of the matrix correspond to MCMC iterations and the columns correspond to the observations in the supplied data (i.e. rows of `X_cont`, `X_cat` and/or `Z_mat`).

`probit_ridgeBART` *Probit ridgeBART for binary outcomes.*

Description

Fit a ridgeBART model of a binary responses using the the Albert & Chib (1993) data augmentation for probit models.

Usage

```
probit_ridgeBART(
  Y_train,
  X_cont_train = matrix(0, nrow = 1, ncol = 1),
  X_cat_train = matrix(0, nrow = 1, ncol = 1),
  Z_mat_train = matrix(0, nrow = 1, ncol = 1),
  X_cont_test = matrix(0, nrow = 1, ncol = 1),
  X_cat_test = matrix(0L, nrow = 1, ncol = 1),
  Z_mat_test = matrix(0, nrow = 1, ncol = 1),
  unif_cuts = rep(TRUE, times = ncol(X_cont_train)),
  cutpoints_list = NULL,
  cat_levels_list = NULL,
  sparse = FALSE, p_change = 0.2,
  M = 50, n_bases = 1, activation = "ReLU",
 rho_alpha = 2 * M, rho_nu = 3, rho_lambda = stats::qchisq(.5, df = rho_nu) / rho_nu,
  nd = 1000, burn = 1000, thin = 1,
  save_samples = TRUE, save_trees = TRUE,
  verbose = TRUE, print_every = floor( (nd*thin + burn))/10
)
```

Arguments

`Y_train`	Integer vector of binary (i.e. 0/1) responses for training data
`X_cont_train`	Matrix of continuous predictors for training data. Note, predictors must be re-scaled to lie in the interval [-1,1]. Default is a 1x1 matrix, which signals that there are no continuous predictors in the training data.
`X_cat_train`	Integer matrix of categorical predictors for training data. Note categorical levels should be 0-indexed. That is, if a categorical predictor has 10 levels, the values should run from 0 to 9. Default is a 1x1 matrix, which signals that there are no categorical predictors in the training data.

Z_mat_train	Matrix of smoothing predictors for training data. Note, predictors must be re-scaled to lie in the interval [-1,1]. Default is a 1x1 matrix, which signals that there are no continuous predictors in the training data.
X_cont_test	Matrix of continuous predictors for testing data. Default is a 1x1 matrix, which signals that testing data is not provided.
X_cat_test	Integer matrix of categorical predictors for testing data. Default is a 1x1 matrix, which signals that testing data is not provided.
Z_mat_test	Matrix of smoothing predictors for testing data. Note, predictors must be re-scaled to lie in the interval [-1,1]. Default is a 1x1 matrix, which signals that there are no continuous predictors in the training data.
unif_cuts	Vector of logical values indicating whether cutpoints for each continuous predictor should be drawn from a continuous uniform distribution (TRUE) or a discrete set (FALSE) specified in cutpoints_list. Default is TRUE for each variable in X_cont_train
cutpoints_list	List of length ncol(X_cont_train) containing a vector of cutpoints for each continuous predictor. By default, this is set to NULL so that cutpoints are drawn uniformly from a continuous distribution.
cat_levels_list	List of length ncol(X_cat_train) containing a vector of levels for each categorical predictor. If the j-th categorical predictor contains L levels, cat_levels_list[[j]] should be the vector 0:(L-1). Default is NULL, which corresponds to the case that no categorical predictors are available.
sparse	Logical, indicating whether or not to perform variable selection based on a sparse Dirichlet prior rather than uniform prior; see Linero 2018. Default is FALSE.
p_change	Probability of a change proposal in the MCMC. Defaults to 0.2.
M	Number of trees in the ensemble. Default is 50.
n_bases	Number of smoothing bases used in output function of leaf node. Default is 1.
activation	Specifies the activation function used to form the random basis functions. Options are ReLU, cos, or tanh. See ridgeBART for options. Default is ReLU.
rho_alpha	Confidence parameter of DP prior for the length scale. Default is M, the number of trees in the emsemble.
rho_nu	Shape parameter of base measure distribution choosen by rho_option. Default is 3.
rho_lambda	Scale parameter of base measure distribution choosen by rho_option. Default is qchisq(.5, df = rho_nu) / rho_nu.
nd	Number of posterior draws to return. Default is 1000.
burn	Number of MCMC iterations to be treated as "warmup" or "burn-in". Default is 1000.
thin	Number of post-warmup MCMC iteration by which to thin. Default is 1.
save_samples	Logical, indicating whether to return all posterior samples. Default is TRUE. If FALSE, only posterior mean is returned.

save_trees	Logical, indicating whether or not to save a text-based representation of the tree samples. This representation can be passed to predict_ridgeBART to make predictions at a later time. Default is FALSE.
verbose	Logical, inciating whether to print progress to R console. Default is TRUE.
print_every	As the MCMC runs, a message is printed every print_every iterations. Default is floor((nd*thin + burn)/10) so that only 10 messages are printed.

Details

See ridgeBART for activation options. Implements the Albert & Chib (1993) data augmentation strategy for probit regression and models the regression function with a sum-of-trees. The marginal prior of any evaluation of the regression function f(x) is a normal distribution centered at mu0 with standard deviation tau * sqrt(M). As such, for each x, the induced prior for P(Y = 1 | x) places 95% probability on the interval pnorm(mu0 -2 * tau * sqrt(M)), pnorm(mu0 + 2 * tau * sqrt(M)). By default, we set tau = 1/sqrt(M * n_bases) and mu0 = qnorm(mean(Y_train)) to shrink towards the observed mean.

Value

A list containing

prob.train.mean	Vector containing posterior mean of evaluations of regression function on training data.
prob.train	Matrix with nd rows and length(Y_train) columns. Each row corresponds to a posterior sample of the regression function and each column corresponds to a training observation. Only returned if save_samples == TRUE.
prob.test.mean	Vector containing posterior mean of evaluations of regression function on testing data, if testing data is provided.
prob.test	If testing data was supplied, matrix containing posterior samples of the regression function evaluated on the testing data. Structure is similar to that of yhat_train. Only returned if testing data is passed and save_samples == TRUE.
varcounts	Matrix that counts the number of times a variable was used in a decision rule in each MCMC iteration. Structure is similar to that of yhat_train, with rows corresponding to MCMC iteration and columns corresponding to predictors, with continuous predictors listed first followed by categorical predictors
trees	A list (of length M) of change logs and indexed decison rules and leaf parameters containing textual representations of the regression trees. These strings are parsed by predict_ridgeBART to reconstruct the C++ representations of the sampled trees.

References

Albert, J.H. and Chib, S. (1993). Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*. **88**(422):669–679. doi:10.1080/01621459.1993.10476321.

See Also

ridgeBART for continuous outcomes.

---

ridgeBART *BART with ridge function outputs.*

Description

Implements an extension of Bayesian Additive Regression Trees (BART; Chipman et al. 2010) that replaces constant leaf node outputs with linear combinations of ridge functions. These ridge functions are a composition between an affine transformation and a user-specified non-linear activation function (e.g., cosine, hyperbolic tangent, logistic, or ReLU).

Usage

```
ridgeBART(
  Y_train,
  X_cont_train = matrix(0, nrow = 1, ncol = 1),
  X_cat_train = matrix(0, nrow = 1, ncol = 1),
  Z_mat_train = matrix(0, nrow = 1, ncol = 1),
  X_cont_test = matrix(0, nrow = 1, ncol = 1),
  X_cat_test = matrix(0L, nrow = 1, ncol = 1),
  Z_mat_test = matrix(0, nrow = 1, ncol = 1),
  unif_cuts = rep(TRUE, times = ncol(X_cont_train)),
  cutpoints_list = NULL,
  cat_levels_list = NULL,
  sparse = FALSE, p_change = 0.2, sigma0 = 1.0,
  M = 50, n_bases = 1, activation = "ReLU",
 rho_alpha = 2 * M, rho_nu = 3, rho_lambda = stats::qchisq(.5, df = rho_nu) / rho_nu,
  nd = 1000, burn = 1000, thin = 1,
  save_samples = TRUE, save_trees = TRUE,
  verbose = TRUE, print_every = floor( (nd*thin + burn))/10
)
```

Arguments

Y_train	Vector of continuous responses for training data
X_cont_train	Matrix of continuous predictors for training data. Note, predictors must be re-scaled to lie in the interval [-1,1]. Default is a 1x1 matrix, which signals that there are no continuous predictors in the training data.
X_cat_train	Integer matrix of categorical predictors for training data. Note categorical levels should be 0-indexed. That is, if a categorical predictor has 10 levels, the values should run from 0 to 9. Default is a 1x1 matrix, which signals that there are no categorical predictors in the training data.

`Z_mat_train` Matrix of smoothing predictors for training data. Note, predictors must be re-scaled to lie in the interval [-1,1]. Default is a 1x1 matrix, which signals that there are no continuous predictors in the training data.

`X_cont_test` Matrix of continuous predictors for testing data. Default is a 1x1 matrix, which signals that testing data is not provided.

`X_cat_test` Integer matrix of categorical predictors for testing data. Default is a 1x1 matrix, which signals that testing data is not provided.

`Z_mat_test` Matrix of smoothing predictors for testing data. Note, predictors must be re-scaled to lie in the interval [-1,1]. Default is a 1x1 matrix, which signals that there are no continuous predictors in the training data.

`unif_cuts` Vector of logical values indicating whether cutpoints for each continuous predictor should be drawn from a continuous uniform distribution (`TRUE`) or a discrete set (`FALSE`) specified in `cutpoints_list`. Default is `TRUE` for each variable in `X_cont_train`

`cutpoints_list` List of length `ncol(X_cont_train)` containing a vector of cutpoints for each continuous predictor. By default, this is set to `NULL` so that cutpoints are drawn uniformly from a continuous distribution.

`cat_levels_list`
List of length `ncol(X_cat_train)` containing a vector of levels for each categorical predictor. If the j-th categorical predictor contains L levels, `cat_levels_list[[j]]` should be the vector `0:(L-1)`. Default is `NULL`, which corresponds to the case that no categorical predictors are available.

`sparse` Logical, indicating whether or not to perform variable selection based on a sparse Dirichlet prior rather than uniform prior; see Linero 2018. Default is `FALSE`.

`p_change` Probability of a change proposal in the MCMC. Defaults to 0.2.

`sigma0` Prior variance. Defaults to 1.0.

`M` Number of trees in the ensemble. Default is 50.

`n_bases` Number of smoothing bases used in output function of leaf node. Default is 1.

`activation` Specifies the activation function used to form the random basis functions. Options are `ReLU`, `cos`, or `tanh`. See details for further description. Default is `ReLU`.

`rho_alpha` Concentration parameter of DP prior for the length scale. Default is `2*M`.

`rho_nu` Shape parameter of base measure distribution choosen by `rho_option`. Default is 3.

`rho_lambda` Scale parameter of base measure distribution choosen by `rho_option`. Default is `qchisq(0.5, df = nu) / nu`.

`nd` Number of posterior draws to return. Default is 1000.

`burn` Number of MCMC iterations to be treated as "warmup" or "burn-in". Default is 1000.

`thin` Number of post-warmup MCMC iteration by which to thin. Default is 1.

`save_samples` Logical, indicating whether to return all posterior samples. Default is `TRUE`. If `FALSE`, only posterior mean is returned.

save_trees	Logical, indicating whether or not to save a text-based representation of the tree samples. This representation can be passed to predict_ridgeBART to make predictions at a later time. Default is FALSE.
verbose	Logical, inciating whether to print progress to R console. Default is TRUE.
print_every	As the MCMC runs, a message is printed every print_every iterations. Default is floor((nd*thin + burn)/10) so that only 10 messages are printed.

Details

Activation options: ReLU: (h(x) = max(x, 0)) cos: (h(x) = sqrt(2) * cos(x)) tanh: (h(x) = tanh(x))

Value

A list containing

y_mean	Mean of the training observations (needed by predict_ridgeBART)
y_sd	Standard deviation of the training observations (needed by predict_ridgeBART)
yhat.train.mean	Vector containing posterior mean of evaluations of regression function on training data.
yhat.train	Matrix with nd rows and length(Y_train) columns. Each row corresponds to a posterior sample of the regression function and each column corresponds to a training observation. Only returned if save_samples == TRUE.
yhat.test.mean	Vector containing posterior mean of evaluations of regression function on testing data, if testing data is provided.
yhat.test	If testing data was supplied, matrix containing posterior samples of the regression function evaluated on the testing data. Structure is similar to that of yhat_train. Only returned if testing data is passed and save_samples == TRUE.
sigma	Vector containing ALL samples of the residual standard deviation, including burn-in.
varcounts	Matrix that counts the number of times a variable was used in a decision rule in each MCMC iteration. Structure is similar to that of yhat_train, with rows corresponding to MCMC iteration and columns corresponding to predictors, with continuous predictors listed first followed by categorical predictors
trees	A list (of length M) of change logs and indexed decison rules and leaf parameters containing textual representations of the regression trees. These strings are parsed by predict_ridgeBART to reconstruct the C++ representations of the sampled trees.

References

Chipman, H, George, E.I., and McCulloch, R. (2008) BART: Bayesian Additive Regression Trees. *Annals of Applied Statistics*. **4**(1):266–298. doi:10.1214/09AOAS285.

Yee, R., Ghosh, S, and Deshpande, S.K. (2026+). Scalable piecewise smoothing with BART. *arXiv preprint* arXiv:2411.07984. doi:10.48550/arXiv.2411.07984.

See Also

probit_ridgeBART for binary outcomes.

Examples

```
## Define true regression function (piecewise smooth)
target <- function(x){
  if (x <= -0.5) return(-2 * x)
  else if (x <= 0) return(sin(5 * x))
  else if (x <= 0.5) return((x+1)^2)
  else return(log(x))
}

sigma <- 0.25
n_train <- 2000
n_test <- 100

set.seed(99)
X_train <- matrix(runif(n_train, min = -1, max = 1), ncol = 1)
mu_train <- rep(NA, times = n_train)
for (i in 1:n_train) mu_train[i] <- target(X_train[i,1])
Y_train <- mu_train + rnorm(n_train, mean = 0, sd = sigma)

X_test <- matrix(seq(-1, 1, length.out = n_test), ncol = 1)
mu_test <- rep(NA, times = n_test)
for (i in 1:n_test) mu_test[i] <- target(X_test[i,1])
Y_test <- mu_test + rnorm(n_test, mean = 0, sd = sigma)

## Token run to ensure installation works
fit <-
  ridgeBART(
    Y_train = Y_train,
    X_cont_train = X_train,
    Z_mat_train = X_train,
    X_cont_test = X_test,
    Z_mat_test = X_test,
    save_samples = FALSE, save_trees = FALSE,
    verbose = FALSE,
    nd = 5, burn = 5)

fit <-
  ridgeBART(
    Y_train = Y_train,
    X_cont_train = X_train,
    Z_mat_train = X_train,
    X_cont_test = X_test,
    Z_mat_test = X_test,
    save_samples = FALSE, save_trees = FALSE,
    nd = 100, burn = 100)

## Plot the poster mean regression function evaluations (i.e., fitted values)
```

```
## against the actual values. Points should cluster around 45-degree diagonal
## line y=x

plot(mu_train, fit$yhat.train.mean, pch = 16, cex = 0.5,
  xlab = "Truth", ylab = "Estimate", main = "Training")
abline(a = 0, b = 1, col = 'blue')
```

Index

predict_ridgeBART, 2
probit_ridgeBART, 3, *9*

ridgeBART, *4, 5*, 6, *6*