Package ‘bartXViz’

July 22, 2025

Title Visualization of BART and BARP using SHAP

Version 1.0.6

Date 2025-07-19

Description

Complex machine learning models are often difficult to interpret. Shapley values serve as a powerful tool to understand and explain why a model makes a particular prediction. This package computes variable contributions using permutation-based Shapley values for Bayesian Additive Regression Trees (BART) and its extension with Post-Stratification (BARP). The permutation-based SHAP method proposed by Strumbel and Kononenko (2014) <doi:10.1007/s10115-013-0679-x> is grounded in data obtained via MCMC sampling. Similar to the BART model introduced by Chipman, George, and McCulloch (2010) <doi:10.1214/09-AOAS285>, this package leverages Bayesian posterior samples generated during model estimation, allowing variable contributions to be computed without requiring additional sampling. For XGBoost and baseline adjustments, the approach by Lundberg et al. (2020) <doi:10.1038/s42256-019-0138-9> is also considered.The BARP model proposed by Bisbee (2019) <doi:10.1017/S0003055419000480> extends post-stratification by computing variable contributions within each stratum defined by stratifying variables. The resulting Shapley values are visualized through both global and local explanation methods.

License GPL (>= 2)

Depends R (>= 3.5.0), SuperLearner

Imports bartMachine, BART, ggplot2, ggforce, data.table, ggfittext, ggpubr, foreach, gggenes, Rcpp, dplyr, tidyr, stringr,abind, utils,grid,dbarts, forcats, gridExtra,reshape2, missForest

LinkingTo Rcpp, RcppArmadillo

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation yes

Author Dong-eun Lee [aut, cre],
Eun-Kyung Lee [aut]

Maintainer Dong-eun Lee <`ldongeun.leel@gmail.com`>

Repository CRAN

Date/Publication 2025-07-19 02:30:02 UTC

Contents

`barps` *Bayesian Additive Regression Trees with Post-stratification (BARP)*

Description

This function uses Bayesian Additive Regression Trees (BART) to extrapolate survey data to a level of geographic aggregation at which the original survey was not sampled to be representative of. This is a modified version of the `barp` function from the **BARP** to allow for seed fixation.(`https://github.com/jbisbee1/BARP`)

Usage

```
barps(
  y,
  x,
  dat,
  census,
  geo.unit,
  algorithm = "BARP",
  setSeed = NULL,
  proportion = "None",
  cred_int = c(0.025, 0.975),
  BSSD = FALSE,
  nsims = 200,
  ...
)
```

Arguments

`y`	Outcome of interest. Should be a character of the column name containing the variable of interest.
`x`	Prognostic covariates. Should be a vector of column names corresponding to the covariates used to predict the outcome variable of interest.
`dat`	Survey data containing the x and y column names. The explanatory variables X included in the model must be converted to factors prior to input.
`census`	Census data containing the x column names. It must also have the same structure as X. If the user provides raw census data, BARP will calculate proportions for each unique bin of x covariates. Otherwise, the researcher must calculate bin proportions and indicate the column name that contains the proportions, either as percentages or as raw counts.
`geo.unit`	The column name corresponding to the unit at which outcomes should be aggregated.
`algorithm`	Algorithm for predicting opinions. Can be any algorithm(s) included in the **SuperLearner** package. If multiple algorithms are listed, predicted opinions are provided for each separately, as well as for the weighted ensemble. Defaults to `BARP` which implements Bayesian Additive Regression Trees via `bartMachine`.
`setSeed`	Seed to control random number generation.
`proportion`	The column name corresponding to the proportions for covariate bins in the Census data. If left to the default `None` value, BARP assumes raw census data and estimates bin proportions automatically.
`cred_int`	A vector giving the lower and upper bounds on the credible interval for the predictions.
`BSSD`	Calculate bootstrapped standard deviation. Defaults to `FALSE` in which case the standard deviation is generated by BART's default.
`nsims`	The number of bootstrap simulations.
`...`	Additional arguments to be passed to bartMachine or SuperLearner.

Value

Returns an object of class ‘BARP‘, containing a list of the following components:

`pred.opn`	A `data.frame` where each row corresponds to the geographic unit of interest and the columns summarize the predicted outcome and the upper and lower bounds for the given credible interval (`cred_int`).
`trees`	A `bartMachine` object.
`risk`	A `data.frame` containing the cross-validation risk for each algorithm and the associated weight used in the ensemble predictions. Only useful when multiple algorithms are used.
`barp.dat`	Data containing the estimates and credible intervals for each observation in the input census dataset.
`setSeed`	The random seed value employed during model estimation using bartMachine.
`proportion`	The number of observations in each combination of features.
`x`	The names of the explanatory variables included in the model.

Source

https://github.com/jbisbee1/BARP

See Also

`barps` is used to implement Bayesian Additive Regression Trees based on the **bartMachine** package. For detailed options, see `https://CRAN.R-project.org/package=bartMachine`.

`barps` also uses the **SuperLearner** package to implement alternative regularizers. For more details, see `https://CRAN.R-project.org/package=SuperLearner`.

`census06` *Census-based Population Proportions for Covariate Bins (2006)*

Description

The data frame has the following components:

- This dataset provides population counts in covariate bins based on the 2006 U.S. Census, Each row represents a unique combination of demographic covariates within a state. A data frame with 2940 rows and 9 variables:

  stateid Numeric identifier for the state

  region Region code

  age Age group (1 = 18-30, 2=31-50, 3= 51-65, 4 =65+)

  gXr Gender and race interaction

  educ Education level (1 = LTHS,2 = HS,3 = Some Coll,4 = Coll+)

  pvote Republican presidential vote share in the previous election

  religcon Proportion of population identifying as religious conservatives

  libcon State-level ideology score (liberal to conservative)

  n Population count for the given covariate bin within the state

References

Bisbee, James. "Barp: Improving mister p using bayesian additive regression trees." American Political Science Review 113.4 (2019): 1060-1065.

`decision_plot` *Decision plot*

Description

The `decision_plot` function is a graph that visualizes how individual features contribute to a model's prediction for a specific observation using Shapley values. It can be used to visualize one or multiple observations.

Usage

```
decision_plot(
  object,
  obs_num,
  title = NULL,
  geo.unit = NULL,
  geo.id = NULL,
  bar_default = TRUE
)
```

Arguments

`object`	Enter the name of the object that contains the model's contributions and results obtained using the Explain function.
`obs_num`	single or multiple observation numbers
`title`	plot title
`geo.unit`	The name of the stratum variable in the BARP model as a character.
`geo.id`	Enter a single value of the stratum variable as a character.
`bar_default`	`bar_default` is an option for adjusting the legend's color scale to fit the window length, and its default value is set to `TRUE`. If plots fail to render in LaTeX documents, it is recommended to set this option to `FALSE`.

Value

`plot_out`	The decision plot for one or multiple observations specified in `obs_num`.

Examples

```
## Friedman data
set.seed(2025)
n = 200
p = 5
X = data.frame(matrix(runif(n * p), ncol = p))
y = 10 * sin(pi* X[ ,1] * X[,2]) +20 * (X[,3] -.5)^2 + 10 * X[ ,4] + 5 * X[,5] + rnorm(n)

## BART model
```

```
model = dbarts::bart (X,y, keeptrees = TRUE,ndpost = 200 )

# prediction wrapper function
pfun <- function (object, newdata) {
predict(object, newdata)
}

# Calculate shapley values
model_exp =  Explain  ( model, X = X,  pred_wrapper =  pfun )

# Single observation
decision_plot(model_exp, obs_num=1 )

#Multiple observation
decision_plot(model_exp, obs_num=10:40 )
```

`Explain` *Approximate Shapley values*

Description

Compute fast (approximate) Shapley values for a set of features using the Monte Carlo algorithm described in Strumbelj and Igor (2014). An efficient algorithm for tree-based models, commonly referred to as Tree SHAP, is also supported for **lightgbm**(https://cran.r-project.org/package=lightgbm) and **xgboost**(https://cran.r-project.org/package=xgboost) models; see Lundberg et. al. (2020) for details.

Usage

```
Explain(object, ...)

## Default S3 method:
Explain(
  object,
  feature_names = NULL,
  X = NULL,
  nsim = 1,
  pred_wrapper = NULL,
  newdata = NULL,
  parallel = FALSE,
  ...
)

## S3 method for class 'lm'
Explain(
  object,
  feature_names = NULL,
```

```
  X,
  nsim = 1,
  pred_wrapper,
  newdata = NULL,
  exact = FALSE,
  parallel = FALSE,
  ...
)

## S3 method for class 'xgb.Booster'
Explain(
  object,
  feature_names = NULL,
  X = NULL,
  nsim = 1,
  pred_wrapper,
  newdata = NULL,
  exact = FALSE,
  parallel = FALSE,
  ...
)

## S3 method for class 'lgb.Booster'
Explain(
  object,
  feature_names = NULL,
  X = NULL,
  nsim = 1,
  pred_wrapper,
  newdata = NULL,
  exact = FALSE,
  parallel = FALSE,
  ...
)
```

Arguments

`object`	A fitted model object (e.g., a `ranger::ranger()`, or `xgboost::xgboost()`,object, to name a few).
`...`	Additional arguments to be passed
`feature_names`	Character string giving the names of the predictor variables (i.e., features) of interest. If `NULL`(default) they will be taken from the column names of `X`.
`X`	A matrix-like R object (e.g., a data frame or matrix) containing ONLY the feature columns from the training data (or suitable background data set). If the input includes categorical variables that need to be one-hot encoded, please input data that has been processed using `data.table::one_hot()`. In XGBoost, the input should be the raw dataset containing only the explanatory variables,

	not the data created using `xgb.DMatrix`. **NOTE:** This argument is required whenever `exact = FALSE`.
`nsim`	The number of Monte Carlo repetitions to use for estimating each Shapley value (only used when `exact = FALSE`). Default is `1`. **NOTE:** To obtain the most accurate results, `nsim` should be set as large as feasibly possible.
`pred_wrapper`	Prediction function that requires two arguments, `object` and `newdata`. **NOTE:** This argument is required whenever `exact = FALSE`. The output of this function should be determined according to: **Regression** A numeric vector of predicted outcomes. **Binary classification** A vector of predicted class probabilities for the reference class. **Multiclass classification** A vector of predicted class probabilities for the reference class.
`newdata`	A matrix-like R object (e.g., a data frame or matrix) containing ONLY the feature columns for the observation(s) of interest; that is, the observation(s) you want to compute explanations for. Default is `NULL` which will produce approximate Shapley values for all the rows in `X` (i.e., the training data). If the input includes categorical variables that need to be one-hot encoded, please input data that has been processed using `data.table::one_hot()`.
`parallel`	Logical indicating whether or not to compute the approximate Shapley values in parallel across features; default is `FALSE`. **NOTE:** setting `parallel = TRUE` requires setting up an appropriate (i.e., system-specific) *parallel backend* as described in the **foreach**(`https://cran.r-project.org/package=foreach`); for details, see `vignette("foreach", package = "foreach")` in R.
`exact`	Logical indicating whether to compute exact Shapley values. Currently only available for `stats::lm()`(`https://CRAN.R-project.org/package=STAT`), `xgboost::xgboost()`(`htt` `//CRAN.R-project.org/package=xgboost`), and `lightgbm::lightgbm()`(`https://CRAN.R-project.org/package=lightgbm`) objects. Default is `FALSE`. Note that setting `exact = TRUE` will return explanations for each of the `stats::terms()` in an `stats::lm()` object. Default is `FALSE`.

Value

An object of class `Explain` with the following components :

`newdata`	The data frame formatted dataset employed for the estimation of Shapley values. If a variable has categories, categorical variables are one-hot encoded.
`phis`	A list format containing Shapley values for individual variables.
`fnull`	The expected value of the model's predictions.
`fx`	The prediction value for each observation.
`factor_names`	The name of the categorical variable. If the data contains only continuous or dummy variables, it is set to `NULL`.

Note

Setting `exact = TRUE` with a linear model (i.e., an `stats::lm()` or `stats::glm()` object) assumes that the input features are independent.

References

Strumbelj, E., and Igor K. (2014). Explaining prediction models and individual predictions with feature contributions. Knowledge and information systems, 41(3), 647-665.

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, Su-In (2020). From local explanations to global understanding with Explainable AI for trees. Nature Machine Intelligence, 2(1), 2522–5839.

Examples

```
#
# A projection pursuit regression (PPR) example
#

# Load the sample data; see datasets::mtcars for details
data(mtcars)

# Fit a projection pursuit regression model
fit <- ppr(mpg ~ ., data = mtcars, nterms = 5)

# Prediction wrapper
pfun <- function(object, newdata) {  # needs to return a numeric vector
  predict(object, newdata = newdata)
}

# Compute approximate Shapley values using 10 Monte Carlo simulations
set.seed(101)  # for reproducibility
shap <- Explain(fit, X = subset(mtcars, select = -mpg), nsim = 10,
                pred_wrapper = pfun)
```

`Explain.barp` *Approximate Shapley values computed from the BARP model*

Description

This function is implemented to calculate the contribution of each variable in the BARP (Bayesian Additive Regression Tree with post-stratification) model using the permutation method.

Usage

```
## S3 method for class 'barp'
Explain(
  object,
  feature_names = NULL,
  X = NULL,
  nsim = 1,
  pred_wrapper = NULL,
```

```
  census = NULL,
  geo.unit = NULL,
  parallel = FALSE,
  ...
)
```

Arguments

`object`	A BARP model (Bayesian Additive Regression Tree) estimated using the `barps` function, a modified version of the `barp` function from the BARP library with a fixed seed.
`feature_names`	The name of the variable for which you want to check the contribution. The default value is set to `NULL`, which means the contribution of all variables in `X` will be calculated.
`X`	The dataset containing all independent variables used as input when estimating the BART model. The explanatory variables `X` included in the model must be converted to factors prior to input.
`nsim`	The number of Monte Carlo sampling iterations, which is fixed at `1` by default in the case of the BARP model.
`pred_wrapper`	A function used to estimate the predicted values of the model.
`census`	Census data containing the names of the `X` columns. It should also have the same format as `X` and include a variable named ‘proportion‘, which indicates the number of individuals corresponding to each combination.
`geo.unit`	Enter the name of the stratification variable used in post stratification.
`parallel`	The default value is set to `FALSE`, but it can be changed to `TRUE` for parallel computation.
`...`	Additional arguments to be passed

Value

Returns of class `Explainbarp` with consisting of a list with the following components:

`phis`	A list containing the Shapley values for each variable.
`newdata`	The data used to check the contribution of variables. If a variable has two categories, it is dummy-coded, and if it has three or more categories, categorical variables are one-hot encoded.
`fnull`	The expected value of the model’s predictions.
`fx`	The prediction value for each observation.
`factor_names`	The name of the categorical variable. If the data contains only continuous or dummy variables, it is set to `NULL`.

`Explain.bart` *Approximate Shapley values computed from a BART model fitted using* `bart`

Description

'Explain.bart' function is used to calculate the contribution of each variable in the Bayesian Additive Regression Trees (BART) model using permutation. It is used to compute the Shapley values of models estimated using the `bart` function from the `dbarts`.

Usage

```
## S3 method for class 'bart'
Explain(
  object,
  feature_names = NULL,
  X = NULL,
  nsim = 1,
  pred_wrapper = NULL,
  newdata = NULL,
  parallel = FALSE,
  ...
)
```

Arguments

`object`	A BART model (Bayesian Additive Regression Tree) estimated using the `bart` function from the **dbarts**.
`feature_names`	The name of the variable for which you want to check the contribution. The default value is set to `NULL`, which means the contribution of all variables in `X` will be calculated.
`X`	The dataset containing all independent variables used as input when estimating the BART model.
`nsim`	The number of Monte Carlo sampling iterations, which is fixed at `1` by default in the case of the BART model.
`pred_wrapper`	A function used to estimate the predicted values of the model.
`newdata`	New data containing the variables included in the model. This is used when checking the contribution of newly input data using the model. The default value is set to `NULL`, meaning that the input `X` data, i.e., the data used for model estimation, will be used by default.
`parallel`	The default value is set to `FALSE`, but it can be changed to `TRUE` for parallel computation.
`...`	Additional arguments to be passed

Value

Returns of class `ExplainBART` with consisting of a list with the following components:

`phis`	A list containing the Shapley values for each variable.
`newdata`	The data used to check the contribution of variables. If a variable has categories, categorical variables are one-hot encoded.
`fnull`	The expected value of the model's predictions.
`fx`	The prediction value for each observation.
`factor_names`	The name of the categorical variable. If the data contains only continuous or dummy variables, it is set to `NULL`.

Examples

```
## Friedman data
set.seed(2025)
n = 200
p = 5
X = data.frame(matrix(runif(n * p), ncol = p))
y = 10 * sin(pi* X[ ,1] * X[,2]) +20 * (X[,3] -.5)^2 + 10 * X[ ,4] + 5 * X[,5] + rnorm(n)

## Using the dbarts library
model = dbarts::bart(X,y,keeptrees = TRUE ,  ndpost = 200)

## prediction wrapper function
pfun <- function(object, newdata) {
       predict(object , newdata)
       }

## Calculate shapley values
model_exp =  Explain  ( model, X = X,  pred_wrapper =  pfun )
```

`Explain.bartMachine` *Approximate Shapley values computed from a BART model fitted using* `bartMachine`

Description

This function is used to calculate the contribution of each variable in the Bayesian Additive Regression Trees (BART) model using permutation. It is used to compute the Shapley values of models estimated using the `bartMachine` function from the **bartMachine**.

Usage

```
## S3 method for class 'bartMachine'
Explain(
  object,
```

```
  feature_names = NULL,
  X = NULL,
  nsim = 1,
  pred_wrapper = NULL,
  newdata = NULL,
  parallel = FALSE,
  ...
)
```

Arguments

`object`	A BART model (Bayesian Additive Regression Tree) estimated using the `bartMachine` function from the **bartMachine**.
`feature_names`	The name of the variable for which you want to check the contribution. The default value is set to `NULL`, which means the contribution of all variables in `X` will be calculated.
`X`	The dataset containing all independent variables used as input when estimating the BART model. Categorical or character variables must not contain an underscore ("_") in their values or labels.
`nsim`	The number of Monte Carlo repetitions used for estimating each Shapley value is set to `1` by default for the BART model.
`pred_wrapper`	A function used to estimate the predicted values of the model.
`newdata`	New data containing the variables included in the model. This is used when checking the contribution of newly input data using the model. The default value is set to `NULL`, meaning that the input `X` data, i.e., the data used for model estimation, will be used by default.
`parallel`	The default value is set to `FALSE`, but it can be changed to `TRUE` for parallel computation.
`...`	Additional arguments to be passed

Value

An object of class `ExplainbartMachine` with the following components :

`phis`	A list containing the Shapley values for each variable.
`newdata`	The data used to check the contribution of variables. If a variable has categories, categorical variables are one-hot encoded.
`fnull`	The expected value of the model's predictions.
`fx`	The prediction value for each observation.
`factor_names`	The name of the categorical variable. If the data contains only continuous or dummy variables, it is set to `NULL`.

Examples

```
## Friedman data
set.seed(2025)
```

```
n = 200
p = 5
X = data.frame(matrix(runif(n * p), ncol = p))
y = 10 * sin(pi* X[ ,1] * X[,2]) +20 * (X[,3] -.5)^2 + 10 * X[ ,4] + 5 * X[,5] + rnorm(n)

##  Using the bartMachine library
model = bartMachine::bartMachine(X,y, seed = 2025, num_iterations_after_burn_in =200 )

## prediction wrapper function
pfun <- function (object, newdata) {
  bartMachine::bart_machine_get_posterior(object,newdata) $ y_hat_posterior_samples
  }

## Calculate shapley values
model_exp =  Explain  ( model, X = X,  pred_wrapper =  pfun )
```

`Explain.wbart` *Approximate Shapley values computed from a BART model fitted using* `wbart` *or* `gbart`

Description

`Explain.wbart` function is used to calculate the contribution of each variable in the Bayesian Additive Regression Trees (BART) model using permutation. It is used to compute the Shapley values of models estimated using the `wbart` or `gbart` functions from **BART**.

Usage

```
## S3 method for class 'wbart'
Explain(
  object,
  feature_names = NULL,
  X = NULL,
  nsim = 1,
  pred_wrapper = NULL,
  newdata = NULL,
  parallel = FALSE,
  ...
)
```

Arguments

`object`	A BART model (Bayesian Additive Regression Tree) estimated using the `bart` function from the **dbarts**.
`feature_names`	The name of the variable for which you want to check the contribution. The default value is set to `NULL`, which means the contribution of all variables in `X` will be calculated.

`X`	The dataset containing all independent variables used as input when estimating the BART model.
`nsim`	The number of Monte Carlo repetitions used for estimating each Shapley value is set to `1` by default for the BART model.
`pred_wrapper`	A function used to estimate the predicted values of the model.
`newdata`	New data containing the variables included in the model. This is used when checking the contribution of newly input data using the model. The default value is set to `NULL`, meaning that the input `X` data, i.e., the data used for model estimation, will be used by default.
`parallel`	The default value is set to `FALSE`, but it can be changed to `TRUE` for parallel computation.
`...`	Additional arguments to be passed

Value

Returns of class `ExplainBART` with consisting of a list with the following components:

`phis`	A list containing the Shapley values for each variable.
`newdata`	The data used to check the contribution of variables. If a variable has categories, categorical variables are one-hot encoded.
`fnull`	The expected value of the model's predictions.
`fx`	The prediction value for each observation.
`factor_names`	The name of the categorical variable. If the data contains only continuous or dummy variables, it is set to `NULL`.

Examples

```
## Friedman data
set.seed(2025)
n = 200
p = 5
X = data.frame(matrix(runif(n * p), ncol = p))
y = 10 * sin(pi* X[ ,1] * X[,2]) +20 * (X[,3] -.5)^2 + 10 * X[ ,4] + 5 * X[,5] + rnorm(n)

## Using the BART library
model = BART::wbart(X,y,ndpost=200)
## prediction wrapper function
pfun <- function(object, newdata) {
       predict(object , newdata)
       }

## Calculate shapley values
model_exp =  Explain  ( model, X = X,  pred_wrapper =  pfun )
```

`one_hot` *One Hot Encode*

Description

One-Hot-Encode unordered factor columns of a data.table

Usage

```
one_hot(
  dt,
  cols = "auto",
  sparsifyNAs = FALSE,
  naCols = FALSE,
  dropCols = TRUE,
  dropUnusedLevels = FALSE
)
```

Arguments

`dt`	A data.table
`cols`	Which column(s) should be one-hot-encoded? DEFAULT = "auto" encodes all unordered factor columns
`sparsifyNAs`	Should NAs be converted to 0s?
`naCols`	Should columns be generated to indicate the present of NAs? Will only apply to factor columns with at least one NA
`dropCols`	Should the resulting data.table exclude the original columns which are one-hot-encoded?
`dropUnusedLevels`	Should columns of all 0s be generated for unused factor levels?

Details

One-hot-encoding converts an unordered categorical vector (i.e. a factor) to multiple binarized vectors where each binary vector of 1s and 0s indicates the presence of a class (i.e. level) of the of the original vector.

Value

data.table object From the input data, a data frame in which categorical variables have been one-hot encoded is returned.

Source

https://cran.r-project.org/web/packages/mltools

Examples

```
library(data.table)

dt <- data.table(
  ID = 1:4,
  color = factor(c("red", NA, "blue", "blue"), levels=c("blue", "green", "red"))
)

one_hot(dt)
one_hot(dt, sparsifyNAs=TRUE)
one_hot(dt, naCols=TRUE)
one_hot(dt, dropCols=FALSE)
one_hot(dt, dropUnusedLevels=TRUE)
```

`plot.Explain` *A function for visualizing the Shapley values*

Description

The `plot.Explain` function provides various visualization methods for Shapley values. The values and format used in the graph are determined based on the input parameters.

Usage

```
## S3 method for class 'Explain'
plot(
  x,
  average = NULL,
  type = NULL,
  num_post = NULL,
  plot.flag = TRUE,
  adjust = FALSE,
  probs = 0.95,
  title = NULL,
  ...
)
```

Arguments

`x`	An `Explain` class object containing the Shapley values of models.
`average`	Input the reference value for calculating the mean of the object's `phi` list. `"obs"` represents the average based on observations (#post by #variable), while `"post"` represents the average based on posterior samples (#obs by #variable). If `"both"` is entered, calculations are performed based on both observation and posterior sample criteria.

`type`	`"bar"` represents a bar chart that includes the average contribution of each variable, while `"bee"` represents a summary plot, allowing you to determine the graph's format.
`num_post`	To check the contribution of variables for a single posterior sample, enter a value within the number of posterior samples.
`plot.flag`	If `average = "obs"`, the quantile interval of each variable's is provided by default.
`adjust`	The default value is `FALSE`. Enter `TRUE` to check the Shapley values adjusted based on the model's average contribution.
`probs`	Enter the probability for the quantile interval. The default value is `0.95`.
`title`	The title of the plot, with a default value of `NULL`.
`...`	Additional arguments to be passed

Value

The plot is returned based on the specified option.:

`out`	If average is `"obs"` or `"post"`, a bar plot or summary plot is generated based on the selected averaging criterion. If average is `"both"`, a boxplot is displayed to show the distribution of Shapley values computed using both criteria. If adjust is `TRUE`, the adjusted Shapley values are displayed. If `num_post` is specified, a bar plot or summary plot for the selected posterior sample is generated.

`plot.Explainbarp` *Visualization of Shapley values from the BARP model*

Description

This function is implemented to visualize the computed Shapley values in various ways for objects of the `Explainbarp` class. The type of plot generated depends on the input parameters. Since the BARP model is designed to be visualized for a single stratum, the user must specify both the stratum variable and the value of the stratum to be visualized.

Usage

```
## S3 method for class 'Explainbarp'
plot(
  x,
  average = NULL,
  type = NULL,
  num_post = NULL,
  plot.flag = TRUE,
  adjust = FALSE,
  probs = 0.95,
  title = NULL,
  geo.unit = NULL,
```

```
    geo.id = NULL,
    ...
)
```

Arguments

`x`	An `ExplainBARP` class object containing the Shapley values of the BARP model.
`average`	Input the reference value for calculating the mean of the object's phi list. `"obs"` represents the average based on observations (#post by #variable), while `"post"` represents the average based on posterior samples (#obs by #variable). If `"both"` is entered, calculations are performed based on both observation and posterior sample criteria.
`type`	`"bar"` represents a bar chart that includes the average contribution of each variable, while `"bee"` represents a summary plot, allowing you to determine the graph's format.
`num_post`	To check the contribution of variables for a single posterior sample, enter a value within the number of posterior samples.
`plot.flag`	If `average = "obs"`, the quantile interval of each variable's is provided by default.
`adjust`	The default value is `FALSE`. Enter `TRUE` to check the Shapley values adjusted based on the model's average contribution.
`probs`	Enter the probability for the quantile interval. The default value is `0.95`.
`title`	The title of the plot, with a default value of `NULL`.
`geo.unit`	Enter the name of the stratification variable used in post stratification.
`geo.id`	Enter one value of interest among the values of the stratification variable.
`...`	Additional arguments to be passed

Value

The plot is returned based on the specified option.:

`out`	If average is `"obs"` or `"post"`, a bar plot or summary plot is generated based on the selected averaging criterion. If average is `"both"`, a boxplot is displayed to show the distribution of Shapley values computed using both criteria. If adjust is `TRUE`, the adjusted Shapley values are displayed. If `num_post` is specified, a bar plot or summary plot for the selected posterior sample is generated.

`plot.ExplainBART` *A function for visualizing the Shapley values of BART models*

Description

The `plot.ExplainBART` function provides various visualization methods for Shapley values. It is designed to visualize `ExplainBART` class objects, which contain Shapley values computed from models estimated using the `bart` function from the **dbarts** or the `wbart`/`gbart` functions from **BART**. The values and format used in the graph are determined based on the input parameters.

Usage

```
## S3 method for class 'ExplainBART'
plot(
  x,
  average = NULL,
  type = NULL,
  num_post = NULL,
  plot.flag = TRUE,
  adjust = FALSE,
  probs = 0.95,
  title = NULL,
  ...
)
```

Arguments

`x`	An `ExplainBART` class object containing the Shapley values of the BART model.
`average`	Input the reference value for calculating the mean of the object's `phi` list. `"obs"` represents abind the average based on observations (#post by #variable), while `"post"` represents the average based on posterior samples (#obs by #variable). If `"both"` is entered, calculations are performed based on both observation and posterior sample criteria.
`type`	`"bar"` represents a bar chart that includes the average contribution of each variable, while `"bee"` represents a summary plot, allowing you to determine the graph's format.
`num_post`	To check the contribution of variables for a single posterior sample, enter a value within the number of posterior samples.
`plot.flag`	If `average` = `"obs"`, the quantile interval of each variable's is provided by default.
`adjust`	The default value is `FALSE`. Enter `TRUE` to check the Shapley values adjusted based on the model's average contribution.
`probs`	Enter the probability for the quantile interval. The default value is `0.95`.
`title`	The title of the plot, with a default value of `NULL`.
`...`	Additional arguments to be passed

Value

The plot is returned based on the specified option.:

`out`	If average is `"obs"` or `"post"`, a bar plot or summary plot is generated based on the selected averaging criterion. If average is ‘"both"‘, a boxplot is displayed to show the distribution of Shapley values computed using both criteria. If adjust is `TRUE`, the adjusted Shapley values are displayed. If `num_post` is specified, a bar plot or summary plot for the selected posterior sample is generated.

Examples

```
## Friedman data
set.seed(2025)
n = 200
p = 5
X = data.frame(matrix(runif(n * p), ncol = p))
y = 10 * sin(pi* X[ ,1] * X[,2]) +20 * (X[,3] -.5)^2 + 10 * X[ ,4] + 5 * X[,5] + rnorm(n)

## Using dbarts
model = dbarts::bart (X,y, keeptrees = TRUE, ndpost = 200)

# prediction wrapper function
pfun <- function (object, newdata) {
predict(object, newdata)
}

# Calculate shapley values
model_exp =  Explain  ( model, X = X,  pred_wrapper =  pfun )

# Distribution of Shapley values (boxplot)
# computed based on observation and posterior sample criteria
plot(model_exp,average = "both" )

# Barplot based on observation criteria
plot(model_exp,average = "obs",type ="bar",probs = 0.95)

# Barplot based on posterior sample
plot(model_exp,average = "post",type ="bar" )

# Summary plot based on posterior sample
plot(model_exp,average = "post",type ="bees" )

# Summary plot of the 100th posterior sample
plot(model_exp,average = "post",type ="bees",num_post = 100)

# Barplot of the adjusted baseline
plot(model_exp, type ="bar",  adjust= TRUE )
```

`plot.ExplainbartMachine`

A function for visualizing the Shapley values of BART models

Description

The `plot.ExplainbartMachine` function provides various visualization methods for Shapley values. It is designed to visualize `ExplainbartMachine` class objects, which contain Shapley values computed from models estimated using the `bartMachine` function from the **bartMachine**. The values and format used in the graph are determined based on the input parameters.

Usage

```
## S3 method for class 'ExplainbartMachine'
plot(
  x,
  average = NULL,
  type = NULL,
  num_post = NULL,
  plot.flag = TRUE,
  adjust = FALSE,
  probs = 0.95,
  title = NULL,
  ...
)
```

Arguments

`x`	An `ExplainbartMachine` class object containing the Shapley values of the BART model.
`average`	Input the reference value for calculating the mean of the object's `phi` list. `"obs"` represents abind the average based on observations (#post by #variable), while `"post"` represents the average based on posterior samples (#obs by #variable). If `"both"` is entered, calculations are performed based on both observation and posterior sample criteria.
`type`	`"bar"` represents a bar chart that includes the average contribution of each variable, while `"bee"` represents a summary plot, allowing you to determine the graph's format.
`num_post`	To check the contribution of variables for a single posterior sample, enter a value within the number of posterior samples.
`plot.flag`	If `average = "obs"`, the quantile interval of each variable's is provided by default.
`adjust`	The default value is `FALSE`. Enter `TRUE` to check the Shapley values adjusted based on the model's average contribution.
`probs`	Enter the probability for the quantile interval. The default value is `0.95`.
`title`	The title of the plot, with a default value of `NULL`.
`...`	Additional arguments to be passed

Value

The plot is returned based on the specified option.:

`out`	If average is `"obs"` or `"post"`, a bar plot or summary plot is generated based on the selected averaging criterion. If average is `"both"`, a boxplot is displayed to show the distribution of Shapley values computed using both criteria. If adjust is `TRUE`, the adjusted Shapley values are displayed. If `num_post` is specified, a bar plot or summary plot for the selected posterior sample is generated.

`svy` *Survey Data on Support for Gay Marriage (2006)*

Description

A dataset used for modeling support for gay marriage in the United States, combining individual- and state-level covariates from a 2006 survey.

- A data frame with 5000 rows and 11 variables:

  id Unique observation identifier

  state Two-letter abbreviation for U.S. state

  stateid Numeric identifier for the state

  region Region code

  age Age group (1 = 18-30, 2 = 31-50, 3 = 51-65, 4 = 65+)

  gXr Gender and race interaction

  educ Education level (1 = LTHS,2 = HS,3 = Some Coll,4 = Coll+)

  supp_gaymar Support for gay marriage (0 = oppose, 1 = support)

  pvote Republican presidential vote share in the previous election

  religcon Proportion of population identifying as religious conservatives

  libcon State-level ideology score (liberal to conservative)

References

Bisbee, James. "Barp: Improving mister p using bayesian additive regression trees." American Political Science Review 113.4 (2019): 1060-1065.

`waterfall_plot` *Waterfall plot*

Description

The `waterfall_plot` function is a bar chart that displays the positive and negative contributions across sequential data points, visualizing how each variable's contributions change for a single observation.

Usage

```
waterfall_plot(
  object,
  obs_num,
  title = NULL,
  geo.unit = NULL,
  geo.id = NULL,
  obs_name = NULL
)
```

Arguments

`object`	Enter the name of the object that contains the model's contributions and results obtained using the Explain function.
`obs_num`	observation number (only one)
`title`	plot title
`geo.unit`	The name of the stratum variable in the BARP model as a character.
`geo.id`	Enter a single value of the stratum variable as a character.
`obs_name`	Enter the name of the vector containing observation IDs or names.

Value

The function returns a waterfall plot.

`plot_out`	The waterfall plot of the observation at index `obs_num`.

Examples

```
## Friedman data
set.seed(2025)
n = 200
p = 5
X = data.frame(matrix(runif(n * p), ncol = p))
y = 10 * sin(pi* X[ ,1] * X[,2]) +20 * (X[,3] -.5)^2 + 10 * X[ ,4] + 5 * X[,5] + rnorm(n)

## Using dbarts library
model = dbarts::bart (X,y,  keeptrees = TRUE,ndpost = 200)

# prediction wrapper function
pfun <- function (object, newdata) {
 predict(object, newdata)
 }

# Calculate shapley values
model_exp =  Explain  ( model, X = X,  pred_wrapper =  pfun )

# Waterfall plot of 100th observation
waterfall_plot(model_exp, obs_num=100)
```

Index