Package 'fastshap'

October 13, 2022

Type Package

Title Fast Approximate Shapley Values

Version 0.0.7

Description Computes fast (relative to other implementations) approximate Shapley values for any supervised learning model. Shapley values help to explain the predictions from any black box model using ideas from game theory; see Strumbel and Kononenko (2014) <doi:10.1007/s10115-013-0679-x> for details.

License GPL (>= 2)

URL https://github.com/bgreenwell/fastshap

BugReports https://github.com/bgreenwell/fastshap/issues

Imports abind, ggplot2 (>= 3.3.4), gridExtra, matrixStats, plyr, Rcpp (>= 1.0.1), tibble

Suggests AmesHousing, covr, earth, htmltools, knitr, lightgbm, ranger, reticulate (>= 1.14), rmarkdown, rstudioapi, tinytest, titanic, xgboost

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.1.2

Encoding UTF-8

NeedsCompilation yes

Author Brandon Greenwell [aut, cre] (<https://orcid.org/0000-0002-8120-0084>)

Maintainer Brandon Greenwell <greenwell.brandon@gmail.com>

Repository CRAN

Date/Publication 2021-12-06 10:40:11 UTC

R topics documented:

`autoplot.explain` *Plotting Shapley values*

Description

Construct Shapley-based importance plots or Shap-based dependence plots.

Usage

```
## S3 method for class 'explain'
autoplot(
  object,
  type = c("importance", "dependence", "contribution"),
  feature = NULL,
  num_features = NULL,
  X = NULL,
  feature_values = NULL,
  color_by = NULL,
  smooth = FALSE,
  smooth_color = "red",
  smooth_linetype = "solid",
  smooth_size = 1,
  smooth_alpha = 1,
  row_num = NULL,
  ...
)
```

Arguments

`object`	An object of class `"explain"`.
`type`	Character string specifying which type of plot to construct. Current options are `"importance"` (for Shapley-based variable importance plots), `"dependence"` (for Shapley-based dependence plots), and `"contribution"` (for visualizing the feature contributions to an individual prediction).
`feature`	Character string specifying which feature to use when `type = "dependence"`. If `NULL` (default) the first feature will be used to construct the plot.
`num_features`	Integer specifying the number of variables to plot. Default is `NULL` which will cause all variables to be displayed.
`X`	A matrix-like R object (e.g., a data frame or matrix) containing ONLY the feature columns from the training data.
`feature_values`	A matrix-like R object (e.g., a data frame or matrix) containing the feature values correposnding to the instance being explained. Only used when `type = "dependence"`. **NOTE:** Must contain the same column structure (e.g., column names, order, etc.) as `X`.

color_by — Character string specifying an optional feature column in X to use for coloring whenever type = "dependence".

smooth — Logical indicating whether or not to add a smoother to the scatterplot whenever type = "dependence". Default is TRUE.

smooth_color — The color to use for the smoother whenever smooth = TRUE. The default is "black"; see geom_smooth for details.

smooth_linetype — The type of line to use for the smoother whenever smooth = TRUE. The default is "solid"; see geom_smooth for details.

smooth_size — The size to use for the smoother whenever smooth = TRUE. The default is 1; see geom_smooth for details.

smooth_alpha — The transparency to use for the smoother whenever smooth = TRUE. The default is 1; see geom_smooth for details.

row_num — Integer specifying a single row/instance in object to plot the explanation when type = "contribution". If NULL (the default) the explanation for the first row/instance will be used.

... — Additional optional arguments to be passed on to geom_col (if type = "importance") or geom_point (if type = "dependence").

Value

A "ggplot" object; see ggplot2-package for details.

Examples

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

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

# Fit a projection pursuit regression model
mtcars.ppr <- ppr(mpg ~ ., data = mtcars, nterms = 1)

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

# Shapley-based plots
library(ggplot2)
autoplot(shap)  # Shapley-based importance plot
autoplot(shap, type = "dependence", feature = "wt", X = mtcars)
autoplot(shap, type = "contribution", row_num = 1)  # explain first row of X
```

explain *Fast approximate Shapley values*

Description

Compute fast (approximate) Shapley values for a set of features.

Usage

```
explain(object, ...)

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

## S3 method for class 'lm'
explain(
  object,
  feature_names = NULL,
  X,
  nsim = 1,
  pred_wrapper,
  newdata = NULL,
  exact = FALSE,
  ...
)

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

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

Arguments

object	A fitted model object (e.g., a ranger, xgboost, or earth object, to name a few).
...	Additional optional arguments to be passed on to laply.
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. **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).
adjust	Logical indicating whether or not to adjust the sum of the estimated Shapley values to satisfy the *additivity* (or *local accuracy*) property; that is, to equal the difference between the model's prediction for that sample and the average prediction over all the training data (i.e., X).
exact	Logical indicating whether to compute exact Shapley values. Currently only available for lm, xgboost, and lightgbm objects. Default is FALSE. Note that setting exact = TRUE will return explanations for each of the terms in an lm object.

Value

A tibble with one column for each feature specified in `feature_names` (if `feature_names = NULL`, the default, there will be one column for each feature in `X`) and one row for each observation in `newdata` (if `newdata = NULL`, the default, there will be one row for each observation in `X`).

Note

Setting `exact = TRUE` with a linear model (i.e., an `lm` or `glm` object) assumes that the input features are independent. Also, setting `adjust = TRUE` is experimental and we follow the same approach as in shap.

See Also

You can find more examples (with larger and more realistic data sets) on the **fastshap** GitHub repository: https://github.com/bgreenwell/fastshap.

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 <- lm(mpg ~ ., data = mtcars)

# 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 = predict)
shap

# Compute exact Shapley (i.e., LinearSHAP) values
shap <- explain(fit, exact = TRUE)
shap

# Shapley-based plots
library(ggplot2)
autoplot(shap)  # Shapley-based importance plot
autoplot(shap, type = "dependence", feature = "wt", X = mtcars)
autoplot(shap, type = "contribution", row_num = 1)  # explain first row of X
```

---

force_plot *Additive force plots*

Description

Visualize Shapley values with additive force style layouts from the Python shap package.

Usage

```
force_plot(object, ...)

## S3 method for class 'explain'
force_plot(
  object,
  baseline = NULL,
  feature_values = NULL,
  display = c("viewer", "html"),
  ...
)
```

Arguments

`object`	An object of class `"explain"`.
`...`	Additional optional arguments. (Currently ignored.)
`baseline`	Numeric giving the average prediction across all the training observations. **NOTE:** It is recommended to provide this argument whenever `object` contains approximate Shapley values.
`feature_values`	A matrix-like R object (e.g., a data frame or matrix) containing the corresponding feature values for the explanations in `object`.
`display`	Character string specifying how to display the results. Current options are `"viewer"` (default) and `"html"`. The latter is necessary for viewing the display inside of an **rmarkdown** document.

Details

The resulting plot shows how each feature contributes to push the model output from the baseline prediction (i.e., the average predicted outcome over the entire training set 'X') to the corresponding model output. Features pushing the prediction higher are shown in red, while those pushing the prediction lower are shown in blue.

Note

This function requires additional software to work. In particular, users will need to make sure they have the following software installed:

- Python (>3.6);
- shap (preferably >=0.36.0).

The reticulate package can be used to help make sure you're set up properly with the above dependencies.

It should also be noted that only exact Shapley explanations (i.e., calling `fastshap::explain()` with `exact = TRUE`) satisfy the so-called *additivity property* where the sum of the feature contributions for x must add up to the difference between the corresponding prediction for x and the average of all the training predictions (i.e., the baseline). Consequently, if you don't set `adjust = TRUE` in the call to `explain` before using `fastshap::force_plot()`, the output value displayed on the plot will not make much sense.

References

Lundberg, Scott M, Bala Nair, Monica S Vavilala, Mayumi Horibe, Michael J Eisses, Trevor Adams, David E Liston, et al. 2018. "Explainable Machine-Learning Predictions for the Prevention of Hypoxaemia During Surgery." Nature Biomedical Engineering 2 (10). Nature Publishing Group: 749.

Examples

```
## Not run:
#
# A projection pursuit regression (PPR) example
#

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

# Fit a projection pursuit regression model
mtcars.ppr <- ppr(mpg ~ ., data = mtcars, nterms = 1)

# Compute approximate Shapley values using 10 Monte Carlo simulations
set.seed(101)  # for reproducibility
shap <- explain(mtcars.ppr, X = subset(mtcars, select = -mpg), nsim = 10,
                pred_wrapper = predict, adjust = TRUE)

# Visualize first explanation
preds <- predict(mtcars.ppr, newdata = mtcars)
x <- subset(mtcars, select = -mpg)[1L, ]  # take first row of feature values
force_plot(shap[1L, ], baseline = mean(preds), feature_values = x)

## End(Not run)
```

---

gen_friedman *Friedman benchmark data*

Description

Simulate data from the Friedman 1 benchmark problem. These data were originally described in Friedman (1991) and Breiman (1996). For details, see sklearn.datasets.make_friedman1.

Usage

```
gen_friedman(
  n_samples = 100,
  n_features = 10,
  n_bins = NULL,
  sigma = 0.1,
  seed = NULL
)
```

Arguments

n_samples	Integer specifying the number of samples (i.e., rows) to generate. Default is 100.
n_features	Integer specifying the number of features to generate. Default is 10.
n_bins	Integer specifying the number of (roughly) equal sized bins to split the response into. Default is NULL for no binning. Setting to a positive integer > 1 effectively turns this into a classification problem where n_bins gives the number of classes.
sigma	Numeric specifying the standard deviation of the noise.
seed	Integer specifying the random seed. If NULL (the default) the results will be different each time the function is run.

Value

A data frame of simulated observations from the Friedman 1 benchmark problem.

Note

This function is mostly used for internal testing.

References

Breiman, Leo (1996) Bagging predictors. Machine Learning 24, pages 123-140.

Friedman, Jerome H. (1991) Multivariate adaptive regression splines. The Annals of Statistics 19 (1), pages 1-67.

Examples

```
gen_friedman()
```

Index