## Package ‘fastshap’

February 22, 2024

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Fast Approximate Shapley Values</td>
</tr>
<tr>
<td>Version</td>
<td>0.1.1</td>
</tr>
<tr>
<td>Description</td>
<td>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) <a href="">doi:10.1007/s10115-013-0679-x</a> for details.</td>
</tr>
<tr>
<td>License</td>
<td>GPL (&gt;= 2)</td>
</tr>
<tr>
<td>BugReports</td>
<td><a href="https://github.com/bgreenwell/fastshap/issues">https://github.com/bgreenwell/fastshap/issues</a></td>
</tr>
<tr>
<td>Imports</td>
<td>foreach, Rcpp (&gt;= 1.0.1), utils</td>
</tr>
<tr>
<td>Enhances</td>
<td>lightgbm, xgboost</td>
</tr>
<tr>
<td>Suggests</td>
<td>AmesHousing, covr, earth, knitr, ranger, rmarkdown, shapviz (&gt;= 0.8.0), tibble, tinytest (&gt;= 1.4.1)</td>
</tr>
<tr>
<td>LinkingTo</td>
<td>Rcpp, RcppArmadillo</td>
</tr>
<tr>
<td>RoxygenNote</td>
<td>7.2.3</td>
</tr>
<tr>
<td>Encoding</td>
<td>UTF-8</td>
</tr>
<tr>
<td>Depends</td>
<td>R (&gt;= 3.6.0)</td>
</tr>
<tr>
<td>LazyData</td>
<td>true</td>
</tr>
<tr>
<td>VignetteBuilder</td>
<td>knitr</td>
</tr>
<tr>
<td>NeedsCompilation</td>
<td>yes</td>
</tr>
<tr>
<td>Author</td>
<td>Brandon Greenwell [aut, cre] (<a href="https://orcid.org/0000-0002-8120-0084">https://orcid.org/0000-0002-8120-0084</a>)</td>
</tr>
<tr>
<td>Maintainer</td>
<td>Brandon Greenwell <a href="mailto:greenwell.brandon@gmail.com">greenwell.brandon@gmail.com</a></td>
</tr>
<tr>
<td>Repository</td>
<td>CRAN</td>
</tr>
<tr>
<td>Date/Publication</td>
<td>2024-02-22 22:00:02 UTC</td>
</tr>
</tbody>
</table>
R topics documented:

explain ................................................................. 2
gen_friedman ....................................................... 6
titanic ................................................................. 7
titanic_mice ......................................................... 7

Index

explain Fast 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 and 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,
  adjust = FALSE,
  baseline = NULL,
  shap_only = TRUE,
  parallel = FALSE,
  ...
)

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

baseline = NULL,
shap_only = TRUE,
parallel = FALSE,
...

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

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

Arguments

object A fitted model object (e.g., a ranger::ranger(), xgboost::xgboost(), or earth::earth() object, to name a few).

... Additional optional arguments to be passed on to foreach::foreach() whenever parallel = TRUE. For example, you may need to supply additional packages that the parallel task depends on via the .packages argument to foreach::foreach().

NOTE: foreach::foreach()'s .combine argument is already set internally by explain(), so passing it via the ... argument would likely result in an error.

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). **NOTE:** This argument is required whenever \( \text{exact} = \text{FALSE} \).

\( \text{nsim} \) The number of Monte Carlo repetitions to use for estimating each Shapley value (only used when \( \text{exact} = \text{FALSE} \)). Default is 1. **NOTE:** To obtain the most accurate results, \( \text{nsim} \) should be set as large as feasibly possible.

\( \text{pred\_wrapper} \) Prediction function that requires two arguments, object and \( \text{newdata} \). **NOTE:** This argument is required whenever \( \text{exact} = \text{FALSE} \). 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.

\( \text{newdata} \) 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).

\( \text{adjust} \) Logical indicating whether or not to adjust the sum of the estimated Shapley values to satisfy the **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., \( X \)). Default is \( \text{FALSE} \) and setting to \( \text{TRUE} \) requires \( \text{nsim} > 1 \).

\( \text{baseline} \) Numeric baseline to use when adjusting the computed Shapley values to achieve **local accuracy**. Adjusted Shapley values for a single prediction (\( f(x) \)) will sum to the difference \( f(x) - \text{baseline} \). Defaults to NULL, which corresponds to the average predictions computed from \( X \), and zero otherwise (i.e., no additional predictions will be computed and the baseline attribute of the output will be set to zero).

\( \text{shap\_only} \) Logical indicating whether or not to include additional output useful for plotting (i.e., \( \text{newdata} \) and the \( \text{baseline} \) value.). This is convenient, for example, when using \texttt{shapviz::shapviz()} for plotting. Default is \( \text{TRUE} \).

\( \text{parallel} \) Logical indicating whether or not to compute the approximate Shapley values in parallel across features; default is \( \text{FALSE} \). **NOTE:** setting \( \text{parallel} = \text{TRUE} \) requires setting up an appropriate (i.e., system-specific) **parallel backend** as described in the \texttt{foreach}; for details, see vignette("foreach", package = "foreach") in R.

\( \text{exact} \) Logical indicating whether to compute exact Shapley values. Currently only available for \texttt{stats::lm()}, \texttt{xgboost::xgboost()}, and \texttt{lightgbm::lightgbm()} objects. Default is \( \text{FALSE} \). Note that setting \( \text{exact} = \text{TRUE} \) will return explanations for each of the \texttt{stats::terms()} in an \texttt{stats::lm()} object. Default is \( \text{FALSE} \).
**Value**

If `shap_only = TRUE` (the default), a matrix is returned 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`). Additionally, the returned matrix will have an attribute called "baseline" containing the baseline value. If `shap_only = FALSE`, then a list is returned with three components:

- `shapley_values` - a matrix of Shapley values (as described above);
- `feature_values` - the corresponding feature values (for plotting with `shapviz::shapviz()`);
- `baseline` - the corresponding baseline value (for plotting with `shapviz::shapviz()`).

**Note**

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

**References**


**See Also**

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

**Examples**

```r
# 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)
head(shap)

---

**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**

```r
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.

**Note**

This function is mostly used for internal testing.

**References**


**Examples**

```r
gen_friedman()
```
**titanic**

*Survival of Titanic passengers*

**Description**

A data set containing the survival outcome, passenger class, age, sex, and the number of family members for a large number of passengers aboard the ill-fated Titanic.

**Usage**

`titanic`

**Format**

A data frame with 1309 observations on the following 6 variables:

- **survived** binary with levels "yes" for survived and "no" otherwise;
- **pclass** integer giving the corresponding passenger (i.e., ticket) class with values 1–3;
- **age** the age in years of the corresponding passenger (with 263 missing values);
- **sex** factor giving the sex of each passenger with levels "male" and "female";
- **sibsp** integer giving the number of siblings/spouses aboard for each passenger (ranges from 0–8);
- **parch** integer giving the number of parents/children aboard for each passenger (ranges from 0–9).

**Note**

As mentioned in the column description, age contains 263 NAs (or missing values). For a complete version (or versions) of the data set, see `titanic_mice`.

**Source**

[https://hbiostat.org/data/](https://hbiostat.org/data/)

---

**titanic_mice**

*Survival of Titanic passengers*

**Description**

The `titanic` data set contains 263 missing values (i.e., NAs) in the age column. This version of the data contains imputed values for the age column using *multivariate imputation by chained equations* via the `mice` package. Consequently, this is a list containing 11 imputed versions of the observations contained in the `titanic` data frame; each completed data sets has the same dimension and column structure as `titanic`.
**Usage**

titanic_mice

**Format**

An object of class `mild` (inherits from `list`) of length 21.

**Source**

Index

* datasets
  titanic, 7
  titanic_mice, 7

earth::earth(), 3
explain, 2

foreach::foreach(), 3

gen_friedman, 6

lightgbm::lightgbm(), 4

ranger::ranger(), 3

shapviz::shapviz(), 4, 5
stats::glm(), 5
stats::lm(), 4, 5
stats::terms(), 4

titanic, 7, 7

titanic_mice, 7, 7

xgboost::xgboost(), 3, 4