Package 'kernelshap'

September 5, 2022

Title Kernel SHAP

Version 0.2.0

Description Multidimensional version of the iterative Kernel SHAP algorithm described in Ian Covert and Su-In Lee (2021) <http://proceedings.mlr.press/v130/covert21a>. SHAP values are calculated iteratively until convergence, along with approximate standard errors. The package allows to work with any model that provides numeric predictions of dimension one or higher. Examples include linear regression, logistic regression (logit or probability scale), other generalized linear models, generalized additive models, and neural networks. The package plays well together with meta-learning packages like 'tidymodels', 'caret' or 'mlr3'. Visualizations can be done using the R package 'shapviz'.

License GPL (>= 2)

Depends R (>= 3.2.0)

Encoding UTF-8

RoxygenNote 7.2.1

LazyData true

Imports doRNG, foreach, MASS, stats, utils

Suggests doFuture, testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/mayer79/kernelshap

BugReports https://github.com/mayer79/kernelshap/issues

NeedsCompilation no

Author Michael Mayer [aut, cre],
David Watson [ctb]

Maintainer Michael Mayer <mayermichael79@gmail.com>

Repository CRAN

Date/Publication 2022-09-05 12:50:08 UTC

R topics documented:

`is.kernelshap` *Check for kernelshap*

Description

Is object of class "kernelshap"?

Usage

```
is.kernelshap(object)
```

Arguments

`object`	An R object.

Value

Returns `TRUE` if `object` has "`kernelshap`" among its classes, and `FALSE` otherwise.

Examples

```
fit <- stats::lm(Sepal.Length ~ ., data = iris)
s <- kernelshap(fit, iris[1:2, -1], bg_X = iris[-1])
is.kernelshap(s)
is.kernelshap("a")
```

kernelshap *Kernel SHAP*

Description

Implements a multidimensional version of the Kernel SHAP algorithm explained in detail in Covert and Lee (2021). It is an iterative refinement of the original Kernel SHAP algorithm of Lundberg and Lee (2017). The algorithm is applied to each row in `X`. Its behaviour depends on the number of features p:

- 2 <= p <= 5: Exact Kernel SHAP values are returned. (Exact regarding the given background data.)
- p > 5: Sampling version of Kernel SHAP. The algorithm iterates until Kernel SHAP values are sufficiently accurate. Approximate standard errors of the SHAP values are returned.
- p = 1: Exact Shapley values are returned.

Usage

```
kernelshap(object, ...)

## Default S3 method:
kernelshap(
  object,
  X,
  bg_X,
  pred_fun = stats::predict,
  bg_w = NULL,
  paired_sampling = TRUE,
  m = "auto",
  exact = TRUE,
  tol = 0.01,
  max_iter = 250,
  parallel = FALSE,
  parallel_args = NULL,
  verbose = TRUE,
  ...
)

## S3 method for class 'ranger'
kernelshap(
  object,
  X,
  bg_X,
  pred_fun = function(m, X, ...) stats::predict(m, X, ...)$predictions,
  bg_w = NULL,
  paired_sampling = TRUE,
```

```
  m = "auto",
  exact = TRUE,
  tol = 0.01,
  max_iter = 250,
  verbose = TRUE,
  ...
)

## S3 method for class 'Learner'
kernelshap(
  object,
  X,
  bg_X,
  pred_fun = function(m, X) m$predict_newdata(X)$response,
  bg_w = NULL,
  paired_sampling = TRUE,
  m = "auto",
  exact = TRUE,
  tol = 0.01,
  max_iter = 250,
  verbose = TRUE,
  ...
)
```

Arguments

`object`	Fitted model object.
`...`	Additional arguments passed to `pred_fun(object, X, ...)`.
`X`	A (n x p) matrix, data.frame, tibble or data.table of rows to be explained. Important: The columns should only represent model features, not the response.
`bg_X`	Background data used to integrate out "switched off" features, often a subset of the training data (around 100 to 200 rows) It should contain the same columns as `X`. Columns not in `X` are silently dropped and the columns are arranged into the order as they appear in `X`.
`pred_fun`	Prediction function of the form `function(object, X, ...)`, providing K >= 1 numeric predictions per row. Its first argument represents the model `object`, its second argument a data structure like `X`. (The names of the first two arguments do not matter.) Additional (named) arguments are passed via `...`. The default, `stats::predict`, will work in most cases. Some exceptions (classes "ranger" and mlr3 "Learner") are handled separately. In other cases, the function must be specified manually.
`bg_w`	Optional vector of case weights for each row of `bg_X`.
`paired_sampling`	Logical flag indicating whether to use paired sampling. The default is `TRUE`. This means that with every feature subset S, also its complement is evaluated, which leads to considerably faster convergence.

m	Number of feature subsets S to be evaluated during one iteration. The default, "auto", equals max(trunc(20*sqrt(p)), 5*p), where p is the number of features. For the paired sampling strategy, 2m evaluations are done per iteration.
exact	If TRUE (default) and the number of features p is at most 5, the algorithm will produce exact Kernel SHAP values. In this case, the arguments m, paired_sampling, tol, and max_iter are ignored.
tol	Tolerance determining when to stop. The algorithm keeps iterating until max(sigma_n) / diff(range(beta_n)) < tol, where the beta_n are the SHAP values of a given observation and sigma_n their standard errors. For multidimensional predictions, the criterion must be satisfied for each dimension separately. The stopping criterion uses the fact that standard errors and SHAP values are all on the same scale.
max_iter	If the stopping criterion (see tol) is not reached after max_iter iterations, the algorithm stops.
parallel	If TRUE, use parallel foreach::foreach() to loop over rows to be explained. Must register backend beforehand, e.g. via "doFuture" package, see Readme for an example. Parallelization automatically disables the progress bar.
parallel_args	A named list of arguments passed to foreach::foreach(), see ?foreach::foreach. Ideally, this is NULL (default). Only relevant if parallel = TRUE. Example on Windows: if object is a generalized additive model fitted with package "mgcv", then one might need to set parallel_args = list(.packages = "mgcv").
verbose	Set to FALSE to suppress messages, warnings, and the progress bar.

Details

During each iteration, m feature subsets are evaluated until the worst standard error of the SHAP values is small enough relative to the range of the SHAP values. This stopping criterion was suggested in Covert and Lee (2021). In the multi-output case, the criterion must be fulfilled for each dimension separately until iteration stops.

Value

An object of class "kernelshap" with the following components:

- S: (n x p) matrix with SHAP values or, if the model output has dimension K > 1, a list of K such matrices.
- X: Same as input argument X.
- baseline: A vector of length K representing the average prediction on the background data.
- SE: Standard errors corresponding to S (and organized like S).
- n_iter: Integer vector of length n providing the number of iterations per row of X.
- converged: Logical vector of length n indicating convergence per row of X.

Methods (by class)

- kernelshap(default): Default Kernel SHAP method.

- `kernelshap(ranger)`: Kernel SHAP method for "ranger" models, see Readme for an example.
- `kernelshap(Learner)`: Kernel SHAP method for "mlr3" models, see Readme for an example.

References

1. Ian Covert and Su-In Lee. Improving KernelSHAP: Practical Shapley Value Estimation Using Linear Regression. Proceedings of The 24th International Conference on Artificial Intelligence and Statistics, PMLR 130:3457-3465, 2021.
2. Scott M. Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. Advances in Neural Information Processing Systems 30, 2017.

Examples

```
# Linear regression
fit <- stats::lm(Sepal.Length ~ ., data = iris)
s <- kernelshap(fit, iris[1:2, -1], bg_X = iris)
s

# Multivariate model
fit <- stats::lm(
  as.matrix(iris[1:2]) ~ Petal.Length + Petal.Width + Species, data = iris
)
s <- kernelshap(fit, iris[1:4, 3:5], bg_X = iris)
s

# Matrix input works as well, and pred_fun can be overwritten
fit <- stats::lm(Sepal.Length ~ ., data = iris[1:4])
pred_fun <- function(fit, X) stats::predict(fit, as.data.frame(X))
X <- data.matrix(iris[2:4])
s <- kernelshap(fit, X[1:3, ], bg_X = X, pred_fun = pred_fun)
s

# Logistic regression
fit <- stats::glm(
  I(Species == "virginica") ~ Sepal.Length + Sepal.Width,
  data = iris,
  family = binomial
)

# On scale of linear predictor
s <- kernelshap(fit, iris[1:2], bg_X = iris)
s

# On scale of response (probability)
s <- kernelshap(fit, iris[1:2], bg_X = iris, type = "response")
s
```

ks_extract *Extractor Function*

Description

Function to extract an element of a "kernelshap" object, e.g., the SHAP values "S".

Usage

```
ks_extract(object, ...)

## S3 method for class 'kernelshap'
ks_extract(
  object,
  what = c("S", "X", "baseline", "SE", "n_iter", "converged"),
  ...
)

## Default S3 method:
ks_extract(object, ...)
```

Arguments

object	Object to extract something.
...	Currently unused.
what	Element to extract. One of "S", "X", "baseline", "SE", "n_iter", or "converged".

Value

The corresponding object is returned.

Methods (by class)

- ks_extract(kernelshap): Method for "kernelshap" object.
- ks_extract(default): No default method available.

Examples

```
fit <- stats::lm(Sepal.Length ~ ., data = iris)
s <- kernelshap(fit, iris[1:2, -1], bg_X = iris[-1])
ks_extract(s, what = "S")
```

print.kernelshap *Prints "kernelshap" Object*

Description

Prints "kernelshap" Object

Usage

```
## S3 method for class 'kernelshap'
print(x, compact = FALSE, n = 2L, ...)
```

Arguments

x	An object of class "kernelshap".
compact	Set to TRUE to hide printing the top n SHAP values, standard errors and feature values.
n	Maximum number of rows of SHAP values, standard errors and feature values to print.
...	Further arguments passed from other methods.

Value

Invisibly, the input is returned.

See Also

kernelshap.

Examples

```
fit <- stats::lm(Sepal.Length ~ ., data = iris)
s <- kernelshap(fit, iris[1:3, -1], bg_X = iris[-1])
s
```

Z_exact *List of weighted on/off matrices Z features*

Description

List of weighted on/off matrices Z features

Usage

```
Z_exact
```

Format

A list of matrices

Index