

Package ‘iBreakDown’

August 27, 2019

Title Model Agnostic Instance Level Variable Attributions

Version 0.9.9

Description Model agnostic tool for decomposition of predictions from black boxes.

Supports additive attributions and attributions with interactions.

The Break Down Table shows contributions of every variable to a final prediction.

The Break Down Plot presents variable contributions in a concise graphical way.

This package works for classification and regression models.

It is an extension of the 'breakDown' package (Staniak and Biecek 2018) <doi:10.32614/RJ-2018-072>,

with new and faster strategies for orderings.

It supports interactions in explanations and has interactive visuals (implemented with 'D3.js' library).

The methodology behind is described in the 'iBreakDown' article (Gosiewska and Biecek 2019) <arXiv:1903.11420>

This package is a part of the 'DrWhy.AI' universe (Biecek 2018) <arXiv:1806.08915>.

Depends R (>= 3.0)

License GPL-2

Encoding UTF-8

LazyData true

Imports ggplot2, DALEX

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, randomForest, e1071, xgboost, ranger, nnet, testthat, r2d3, jsonlite, caret

VignetteBuilder knitr

URL <https://ModelOriented.github.io/iBreakDown/>,
<https://github.com/ModelOriented/iBreakDown>

BugReports <https://github.com/ModelOriented/iBreakDown/issues>

NeedsCompilation no

Author Przemyslaw Biecek [aut, cre],
Alicja Gosiewska [aut],
Hubert Baniecki [aut],

Adam Izdebski [aut],
Dariusz Komosinski [ctb]

Maintainer Przemyslaw Biecek <przemyslaw.biecek@gmail.com>

Repository CRAN

Date/Publication 2019-08-26 22:50:02 UTC

R topics documented:

break_down	2
break_down_uncertainty	4
describe	6
local_attributions	8
local_interactions	11
plot.break_down	13
plot.break_down_uncertainty	15
plotD3	17
plotD3.shap	19
print.break_down	21
print.break_down_description	22
print.break_down_uncertainty	22

Index 24

break_down	<i>Model Agnostic Sequential Variable Attributions</i>
------------	--

Description

This function finds Variable Attributions via Sequential Variable Conditioning. It calls either [local_attributions](#) for additive attributions or [local_interactions](#) for attributions with interactions.

Usage

```
break_down(x, ..., interactions = FALSE)

## S3 method for class 'explainer'
break_down(x, new_observation, ...,
           interactions = FALSE)

## Default S3 method:
break_down(x, data, predict_function = predict,
           new_observation, keep_distributions = FALSE, order = NULL,
           label = class(x)[1], ..., interactions = FALSE)
```

Arguments

x	an explainer created with function <code>explain</code> or a model.
...	parameters passed to 'local_*' functions.
interactions	shall interactions be included?
new_observation	a new observation with columns that correspond to variables used in the model.
data	validation dataset, will be extracted from 'x' if it is an explainer.
predict_function	predict function, will be extracted from 'x' if it's an explainer.
keep_distributions	if 'TRUE', then distribution of partial predictions is stored and can be plotted with the generic 'plot()'.
order	if not 'NULL', then it will be a fixed order of variables. It can be a numeric vector or vector with names of variables.
label	name of the model. By default it is extracted from the 'class' attribute of the model.

Value

an object of the 'break_down' class.

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiecek.github.io/PM_VEE

See Also

[local_attributions](#), [local_interactions](#)

Examples

```
library("DALEX")
library("iBreakDown")
# Toy examples, because CRAN angels ask for them
titanic <- na.omit(titanic)
set.seed(1313)
titanic_small <- titanic[sample(1:nrow(titanic), 500), c(1,2,6,9)]
model_titanic_glm <- glm(survived == "yes" ~ gender + age + fare,
                        data = titanic_small, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                             data = titanic_small[,-9],
                             y = titanic_small$survived == "yes")
bd_rf <- break_down(explain_titanic_glm, titanic_small[, ])
bd_rf
plot(bd_rf, max_features = 3)
```

```

## Not run:
library("randomForest")
set.seed(1313)
# example with interaction
# classification for HR data
model <- randomForest(status ~ . , data = HR)
new_observation <- HR_test[1,]

explainer_rf <- explain(model,
                        data = HR[1:1000,1:5],
                        y = HR$status[1:1000])

bd_rf <- break_down(explainer_rf,
                    new_observation)

bd_rf
plot(bd_rf)

```

break_down_uncertainty

Explanation Level Uncertainty of Sequential Variable Attribution

Description

Function ‘break_down_uncertainty()’ calls ‘B’ times the break down algorithm for random orderings. Then it calculated distribution of attributions for these different orderings. Note that the ‘shap()’ function is just a simplified interface to the ‘break_down_uncertainty()’ function with a default value set to ‘B=25’.

Usage

```

break_down_uncertainty(x, ..., keep_distributions = TRUE, B = 10)

## S3 method for class 'explainer'
break_down_uncertainty(x, new_observation, ...,
                       keep_distributions = TRUE, B = 10)

## Default S3 method:
break_down_uncertainty(x, data,
                       predict_function = predict, new_observation, label = class(x)[1],
                       ..., path = NULL, keep_distributions = TRUE, B = 10)

shap(x, ..., B = 25)

```

Arguments

x an explainer created with function [explain](#) or a model.
... other parameters.

keep_distributions	if TRUE then we will keep distribution for predicted values. It's needed by the describe function.
B	number of random paths
new_observation	a new observation with columns that correspond to variables used in the model.
data	validation dataset, will be extracted from 'x' if it is an explainer.
predict_function	predict function, will be extracted from 'x' if it is an explainer.
label	name of the model. By default it's extracted from the 'class' attribute of the model.
path	if specified, then this path will be highlighted on the plot. Use 'average' in order to show an average effect

Value

an object of the 'break_down_uncertainty' class.

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiecek.github.io/PM_VEE

See Also

[break_down](#), [local_attributions](#)

Examples

```
library("DALEX")
library("iBreakDown")
# Toy examples, because CRAN angels ask for them
titanic <- na.omit(titanic)
set.seed(1313)
titanic_small <- titanic[sample(1:nrow(titanic), 500), c(1,2,6,9)]
model_titanic_glm <- glm(survived == "yes" ~ gender + age + fare,
                        data = titanic_small, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_small[,-9],
                              y = titanic_small$survived == "yes")

# there is no explanation level uncertainty linked with additive models
bd_rf <- break_down_uncertainty(explain_titanic_glm, titanic_small[1, ])
bd_rf
plot(bd_rf)

## Not run:
library("randomForest")
set.seed(1313)
```

```

model <- randomForest(status ~ . , data = HR)
new_observation <- HR_test[1,]

explainer_rf <- explain(model,
                        data = HR[1:1000, 1:5])

bd_rf <- break_down_uncertainty(explainer_rf,
                               new_observation)
bd_rf
plot(bd_rf)

# example for regression - apartment prices
# here we do not have interactions
model <- randomForest(m2.price ~ . , data = apartments)
explainer_rf <- explain(model,
                        data = apartments_test[1:1000, 2:6],
                        y = apartments_test$m2.price[1:1000])

bd_rf <- break_down_uncertainty(explainer_rf, apartments_test[1,])
bd_rf
plot(bd_rf)

bd_rf <- break_down_uncertainty(explainer_rf, apartments_test[1,], path = 1:5)
plot(bd_rf)

bd_rf <- break_down_uncertainty(explainer_rf,
                               apartments_test[1,],
                               path = c("floor", "no.rooms", "district",
                                       "construction.year", "surface"))

plot(bd_rf)

bd_rf <- shap(explainer_rf,
              apartments_test[1,])
bd_rf
plot(bd_rf)
plot(bd_rf, show_boxplots = FALSE)

```

describe

Generates Textual Explanations for Predictive Models

Description

Generic function `describe` generates natural language explanations based on `break_down` and `shap` explanations, what enhances their interpretability.

Usage

```
describe(x, nonsignificance_threshold = 0.15, ...)
```

```

## S3 method for class 'break_down'
describe(x, nonsignificance_treshold = 0.15, ...,
  label = NULL, short_description = FALSE, display_values = FALSE,
  display_numbers = FALSE, display_distribution_details = FALSE,
  display_shap = FALSE)

## S3 method for class 'break_down_uncertainty'
describe(x,
  nonsignificance_treshold = 0.15, ..., label = NULL,
  short_description = FALSE, display_values = FALSE,
  display_numbers = FALSE, display_distribution_details = FALSE,
  display_shap = FALSE)

```

Arguments

x	an explanation created with break_down or shap
nonsignificance_treshold	a numeric specifying a treshold for variable importance
...	other arguments
label	a character string describing model's prediction
short_description	a boolean, returns a short description
display_values	a boolean, displays variables' values
display_numbers	a boolean, displays a description containing numerical values
display_distribution_details	a boolean, displays details about the distribution of model's predictions
display_shap	a boolean, adds information about variables' average contribution. Use only with shap explanation.

Details

Function `describe` generates a textual explanations by extracting information from a [break_down](#) or [shap](#) explanation. It makes an argument justifying why the model's prediction is lower or higher, than it's average prediction. The description consists of an introduction, argumenation and summary making use from the claim, support, evidence argumentation structure, as recomended for the World Universities Debating style.

The function first selects one of four different scenarios, due to `nonsignificance_treshold`. The chosen scenario can be one of the following: 1. Model's prediction for the selected instance is significantly higher than the average prediction. 2. Model's prediction is significantly lower. 3. Model's prediction is close to it's average prediction, however there are significant variables counteracting with each other 4. Model's prediction is close to it's average prediction and all the variables are rather nonsignificant. Then an explanation due to the chosen scenario is generated.

Value

A character string of textual explanation

Examples

```

library("DALEX")
library("randomForest")
library("iBreakDown")

titanic <- na.omit(titanic)
model_titanic_rf <- randomForest(survived == "yes" ~ gender + age + class + embarked +
                                fare + sibsp + parch, data = titanic)

explain_titanic_rf <- explain(model_titanic_rf,
                             data = titanic[, -9],
                             y = titanic$survived == "yes",
                             label = "Random Forest v7")

bd_explanation <- break_down(explain_titanic_rf, titanic[1, ], keep_distributions = TRUE)

description <- describe(bd_explanation,
                        label = "the passanger will survive with probability",
                        short_description = FALSE,
                        display_values = TRUE,
                        display_numbers = TRUE,
                        display_distribution_details = FALSE)

description

```

```

library("DALEX")
library("iBreakDown")
titanic <- na.omit(titanic)
model_titanic_glm <- glm(titanic$survived == "yes" ~ age + gender + class + fare + sibsp,
                        data = titanic[, -9], family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic[, -9],
                              y = titanic$survived == "yes",
                              label = "glm")
passanger <- titanic[sample(nrow(titanic), 1) , -9]
shap_glm <- shap(explain_titanic_glm, passanger)
plot(shap_glm)

describe(shap_glm,
        label = "the selected passanger survives with probability",
        display_shap = TRUE,
        display_numbers = TRUE)

```


Description

This function finds Variable attributions via Sequential Variable Conditioning. The complexity of this function is $O(2^p)$. This function works in a similar way to step-up and step-down greedy approximations in function `break_down`. The main difference is that in the first step the order of variables is determined. And in the second step the impact is calculated.

Usage

```
local_attributions(x, ...)

## S3 method for class 'explainer'
local_attributions(x, new_observation,
  keep_distributions = FALSE, ...)

## Default S3 method:
local_attributions(x, data, predict_function = predict,
  new_observation, label = class(x)[1], keep_distributions = FALSE,
  order = NULL, ...)
```

Arguments

x	an explainer created with function <code>explain</code> or a model.
...	other parameters.
new_observation	a new observation with columns that correspond to variables used in the model.
keep_distributions	if 'TRUE', then distribution of partial predictions is stored and can be plotted with the generic 'plot()'.
data	validation dataset, will be extracted from 'x' if it is an explainer.
predict_function	predict function, will be extracted from 'x' if it is an explainer.
label	name of the model. By default it's extracted from the 'class' attribute of the model.
order	if not 'NULL', then it will be a fixed order of variables. It can be a numeric vector or vector with names of variables.

Value

an object of the 'break_down' class.

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiemek.github.io/PM_VEE

See Also

[break_down](#), [local_interactions](#)

Examples

```

library("DALEX")
library("iBreakDown")
# Toy examples, because CRAN angels ask for them
titanic <- na.omit(titanic)
set.seed(1313)
titanic_small <- titanic[sample(1:nrow(titanic), 500), c(1,2,6,9)]
model_titanic_glm <- glm(survived == "yes" ~ gender + age + fare,
                        data = titanic, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_small[,-9],
                              y = titanic_small$survived == "yes")
bd_rf <- local_attributions(explain_titanic_glm, titanic_small[1, ])
bd_rf
plot(bd_rf, max_features = 3)

## Not run:
library("randomForest")
set.seed(1313)
# example with interaction
# classification for HR data
model <- randomForest(status ~ . , data = HR)
new_observation <- HR_test[1,]

explainer_rf <- explain(model,
                       data = HR[1:1000,1:5],
                       y = HR$status[1:1000])

bd_rf <- local_attributions(explainer_rf,
                          new_observation)
bd_rf
plot(bd_rf)
plot(bd_rf, baseline = 0)

# example for regression - apartment prices
# here we do not have interactions
model <- randomForest(m2.price ~ . , data = apartments)
explainer_rf <- explain(model,
                       data = apartments_test[1:1000,2:6],
                       y = apartments_test$m2.price[1:1000])

bd_rf <- local_attributions(explainer_rf,
                          apartments_test[1,])
bd_rf
plot(bd_rf, digits = 1)

bd_rf <- local_attributions(explainer_rf,
                          apartments_test[1,],
                          keep_distributions = TRUE)
plot(bd_rf, plot_distributions = TRUE)

```

local_interactions *Model Agnostic Sequential Variable Attributions with Interactions*

Description

This function implements decomposition of model predictions with identification of interactions. The complexity of this function is $O(2*p)$ for additive models and $O(2*p^2)$ for interactions. This function works in a similar way to step-up and step-down greedy approximations in function 'break-Down::break_down()'. The main difference is that in the first step the order of variables and interactions is determined. And in the second step the impact is calculated.

Usage

```
local_interactions(x, ...)

## S3 method for class 'explainer'
local_interactions(x, new_observation,
  keep_distributions = FALSE, ...)

## Default S3 method:
local_interactions(x, data, predict_function = predict,
  new_observation, label = class(x)[1], keep_distributions = FALSE,
  order = NULL, interaction_preference = 1, ...)
```

Arguments

x	an explainer created with function explain or a model.
...	other parameters.
new_observation	a new observation with columns that correspond to variables used in the model.
keep_distributions	if 'TRUE', then the distribution of partial predictions is stored in addition to the average.
data	validation dataset, will be extracted from 'x' if it's an explainer.
predict_function	predict function, will be extracted from 'x' if it's an explainer.
label	character - the name of the model. By default it's extracted from the 'class' attribute of the model.
order	if not 'NULL', then it will be a fixed order of variables. It can be a numeric vector or vector with names of variables/interactions.
interaction_preference	an integer specifying which interactions will be present in an explanation. The larger the integer, the more frequently interactions will be presented.


```

y = apartments_test$m2.price[1:1000])

new_observation <- apartments_test[1,]

bd_rf <- local_interactions(explainer_rf,
                           new_observation,
                           keep_distributions = TRUE)

bd_rf
plot(bd_rf)
plot(bd_rf, plot_distributions = TRUE)

```

plot.break_down

Plot Generic for Break Down Objects

Description

Displays a waterfall break down plot for objects of 'break_down' class.

Usage

```

## S3 method for class 'break_down'
plot(x, ..., baseline = NA, max_features = 10,
     min_max = NA, vcolors = DALEX::colors_breakdown_drwhy(),
     digits = 3, rounding_function = round, add_contributions = TRUE,
     shift_contributions = 0.05, plot_distributions = FALSE)

```

Arguments

x	an explanation created with break_down
...	other parameters.
baseline	if numeric then vertical line starts in 'baseline'.
max_features	maximal number of features to be included in the plot. default value is 4.
min_max	a range of OX axis. By default 'NA', therefore it will be extracted from the contributions of 'x'. But it can be set to some constants, useful if these plots are to be used for comparisons.
vcolors	named vector with colors.
digits	number of decimal places ('round') or significant digits ('signif') to be used. See the <code>rounding_function</code> argument.
rounding_function	a function to be used for rounding numbers. This should be signif which keeps a specified number of significant digits or round (which is default) to have the same precision for all components.
add_contributions	if TRUE, variable contributions will be added to the plot

`shift_contributions`
 number describing how much labels should be shifted to the right, as a fraction of range. By default equal to 0.05.

`plot_distributions`
 if 'TRUE' then distributions of conditional proportions will be plotted. This requires `keep_distributions=TRUE` in the `break_down`, `local_attributions`, or `local_interactions`.

Value

a 'ggplot2' object.

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiemek.github.io/PM_VEE

Examples

```
library("DALEX")
library("iBreakDown")
# Toy examples, because CRAN angels ask for them
titanic <- na.omit(titanic)
set.seed(1313)
titanic_small <- titanic[sample(1:nrow(titanic), 500), c(1,2,6,9)]
model_titanic_glm <- glm(survived == "yes" ~ gender + age + fare,
  data = titanic_small, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
  data = titanic_small[,-9],
  y = titanic_small$survived == "yes")
bd_rf <- break_down(explain_titanic_glm, titanic_small[1, ])
bd_rf
plot(bd_rf, max_features = 3)

## Not run:
library("randomForest")
set.seed(1313)
# example with interaction
# classification for HR data
model <- randomForest(status ~ . , data = HR)
new_observation <- HR_test[1,]

explainer_rf <- explain(model,
  data = HR[1:1000,1:5],
  y = HR$status[1:1000])

bd_rf <- local_attributions(explainer_rf,
  new_observation)
bd_rf
plot(bd_rf)
plot(bd_rf, baseline = 0)
```

```

plot(bd_rf, min_max = c(0,1))

bd_rf <- local_attributions(explainer_rf,
                           new_observation,
                           keep_distributions = TRUE)

bd_rf
plot(bd_rf, plot_distributions = TRUE)

bd_rf <- local_interactions(explainer_rf,
                           new_observation,
                           keep_distributions = TRUE)

bd_rf
plot(bd_rf)
plot(bd_rf, plot_distributions = TRUE)

# example for regression - apartment prices
# here we do not have interactions
model <- randomForest(m2.price ~ . , data = apartments)
explainer_rf <- explain(model,
                       data = apartments_test[1:1000,2:6],
                       y = apartments_test$m2.price[1:1000])

bd_rf <- local_attributions(explainer_rf,
                           apartments_test[1,])

bd_rf
plot(bd_rf, digits = 1)
plot(bd_rf, digits = 1, baseline = 0)

bd_rf <- local_attributions(explainer_rf,
                           apartments_test[1,],
                           keep_distributions = TRUE)

plot(bd_rf, plot_distributions = TRUE)

bd_rf <- local_interactions(explainer_rf,
                           new_observation = apartments_test[1,],
                           keep_distributions = TRUE)

bd_rf
plot(bd_rf)
plot(bd_rf, plot_distributions = TRUE)

```

plot.break_down_uncertainty

Plot Generic for Break Down Uncertainty Objects

Description

Plot Generic for Break Down Uncertainty Objects

Usage

```
## S3 method for class 'break_down_uncertainty'
plot(x, ...,
     vcolors = DALEX::colors_breakdown_drwhy(), show_boxplots = TRUE)
```

Arguments

x an explanation created with `break_down_uncertainty`

... other parameters.

vcolors named vector with colors.

show_boxplots logical if 'TRUE' (default) boxplot will be plotted to show uncertainty of attributions

Value

a 'ggplot2' object.

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiecek.github.io/PM_VEE

Examples

```
library("DALEX")
library("iBreakDown")
# Toy examples, because CRAN angels ask for them
titanic <- na.omit(titanic)
set.seed(1313)
titanic_small <- titanic[sample(1:nrow(titanic), 500), c(1,2,6,9)]
model_titanic_glm <- glm(survived == "yes" ~ gender + age + fare,
                        data = titanic_small, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_small[,-9],
                              y = titanic_small$survived == "yes")
bd_rf <- shap(explain_titanic_glm, titanic_small[1, ])
bd_rf
plot(bd_rf)

## Not run:
library("randomForest")
set.seed(1313)

model <- randomForest(status ~ . , data = HR)
new_observation <- HR_test[1,]

explainer_rf <- explain(model,
                       data = HR[1:1000,1:5],
                       y = HR$status[1:1000])
```



```

bd_rf <- break_down_uncertainty(explainer_rf,
                               new_observation,
                               path = c(3,2,4,1,5))

bd_rf
plot(bd_rf)

# example for regression - apartment prices
# here we do not have interactions
model <- randomForest(m2.price ~ . , data = apartments)
explainer_rf <- explain(model,
                        data = apartments_test[1:1000,2:6],
                        y = apartments_test$m2.price[1:1000])

bd_rf <- break_down_uncertainty(explainer_rf,
                               apartments_test[1,],
                               path = c("floor", "no.rooms", "district",
                                         "construction.year", "surface"))

bd_rf
plot(bd_rf)

bd_rf <- shap(explainer_rf,
              apartments_test[1,])

bd_rf
plot(bd_rf)
plot(bd_rf, show_boxplots = FALSE)

```

plotD3

Plot Break Down Objects in D3 with r2d3 package.

Description

Plots waterfall break down for objects of the ‘break_down’ class.

Usage

```
plotD3(x, ...)
```

```

## S3 method for class 'break_down'
plotD3(x, ..., baseline = NA, max_features = 10,
       digits = 3, rounding_function = round, bar_width = 12,
       margin = 0.2, scale_height = FALSE, min_max = NA, vcolors = NA,
       chart_title = NA, time = 0)

```

Arguments

x an explanation created with `break_down`
... other parameters.

baseline	if numeric then vertical line will start in baseline.
max_features	maximal number of features to be included in the plot. Default value is 10.
digits	number of decimal places (round) or significant digits (signif) to be used. See the rounding_function argument.
rounding_function	a function to be used for rounding numbers. This should be <code>signif</code> which keeps a specified number of significant digits or <code>round</code> (which is default) to have the same precision for all components.
bar_width	width of bars in px. By default 12px
margin	extend x axis domain range to adjust the plot. Usually value between 0.1 and 0.3, by default it's 0.2
scale_height	if TRUE, the height of the plot scales with window size
min_max	a range of OX axis. By default 'NA' therefore will be extracted from the contributions of 'x'. But can be set to some constants, usefull if these plots are used for comparisons.
vcolors	If NA (default), DrWhy colors are used.
chart_title	a character. Set custom title
time	in ms. Set animation length

Value

a 'r2d3' object.

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiecek.github.io/PM_VEE

Examples

```
library("DALEX")
library("iBreakDown")

titanic <- na.omit(titanic)
set.seed(1313)
titanic_small <- titanic[sample(1:nrow(titanic), 500), c(1,2,6,9)]
model_titanic_glm <- glm(survived == "yes" ~ gender + age + fare,
                        data = titanic_small, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                             data = titanic_small[,-9],
                             y = titanic_small$survived == "yes",
                             label = "glm")
bd_glm <- local_attributions(explain_titanic_glm, titanic_small[1, ])
bd_glm
plotD3(bd_glm)

library(randomForest)
```

```

m_rf <- randomForest(status ~ . , data = HR[2:2000,])
new_observation <- HR_test[1,]
new_observation

p_fun <- function(object, newdata){predict(object, newdata=newdata, type = "prob")}

bd_rf <- local_attributions(m_rf,
                           data = HR_test,
                           new_observation = new_observation,
                           predict_function = p_fun)

bd_rf
plotD3(bd_rf)

```

plotD3.shap	<i>Plot Shap (Break Down Uncertainty) Objects in D3 with r2d3 package.</i>
-------------	--

Description

Plots shapley values.

Usage

```

## S3 method for class 'shap'
plotD3(x, ..., baseline = NA, max_features = 10,
       digits = 3, rounding_function = round, bar_width = 12,
       margin = 0.2, scale_height = FALSE, min_max = NA, vcolors = NA,
       chart_title = NA, time = 0)

```

Arguments

x	an explanation created with shap
...	other parameters.
baseline	if numeric then vertical line will start in baseline.
max_features	maximal number of features to be included in the plot. Default is 10.
digits	number of decimal places (round) or significant digits (signif) to be used. See the rounding_function argument.
rounding_function	a function to be used for rounding numbers. This should be signif which keeps a specified number of significant digits or round (which is default) to have the same precision for all components.
bar_width	width of bars in px. Default is 12px
margin	extend x axis domain range to adjust the plot. Usually value between 0.1 and 0.3, by default it's 0.2

scale_height	if TRUE, the height of the plot scales with window size.
min_max	a range of OX axis. By default 'NA' therefore will be extracted from the contributions of 'x'. But can be set to some constants, useful if these plots are used for comparisons.
vcolors	If NA (default), DrWhy colors are used.
chart_title	a character. Set custom title
time	in ms. Set animation length

Value

a 'r2d3' object.

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiemek.github.io/PM_VEE

Examples

```
library("DALEX")
library("iBreakDown")

titanic <- na.omit(titanic)
set.seed(1313)
titanic_small <- titanic[sample(1:nrow(titanic), 500),]
model_titanic_glm <- glm(survived == "yes" ~ gender + age + fare + class +
  embarked + country + sibsp + parch,
  data = titanic_small, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
  data = titanic_small[,-9],
  y = titanic_small$survived == "yes",
  label = "glm")
s_glm <- shap(explain_titanic_glm, titanic_small[1, ])
s_glm
plotD3(s_glm)

library(randomForest)

HR_small <- HR[2:500,]
m_rf <- randomForest(status ~. , data = HR_small)
new_observation <- HR_test[1,]
new_observation

p_fun <- function(object, newdata){predict(object, newdata=newdata, type = "prob")}

s_rf <- shap(m_rf,
  data = HR_small[,-6],
  new_observation = new_observation,
  predict_function = p_fun)

plotD3(s_rf, time = 500)
```

print.break_down	<i>Print Generic for Break Down Objects</i>
------------------	---

Description

Print Generic for Break Down Objects

Usage

```
## S3 method for class 'break_down'  
print(x, ..., digits = 3,  
      rounding_function = round)
```

Arguments

x	an explanation created with break_down
...	other parameters.
digits	number of decimal places (round) or significant digits (signif) to be used. See the rounding_function argument.
rounding_function	a function to be used for rounding numbers. This should be signif which keeps a specified number of significant digits or round (which is default) to have the same precision for all components.

Value

a data frame

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiecek.github.io/PM_VEE

```
print.break_down_description
```

Print Generic for Break Down Objects

Description

Print Generic for Break Down Objects

Usage

```
## S3 method for class 'break_down_description'  
print(x, ...)
```

Arguments

x	a description of 'break_down_description' class.
...	other parameters.

Value

a string

```
print.break_down_uncertainty
```

Print Generic for Break Down Uncertainty Objects

Description

Print Generic for Break Down Uncertainty Objects

Usage

```
## S3 method for class 'break_down_uncertainty'  
print(x, ...)
```

Arguments

x	an explanation created with break_down_uncertainty
...	other parameters.

Value

a data frame.

References

Predictive Models: Visual Exploration, Explanation and Debugging https://pbiemek.github.io/PM_VEE

Examples

```
library("DALEX")
library("iBreakDown")
# Toy examples, because CRAN angels ask for them
titanic <- na.omit(titanic)
set.seed(1313)
titanic_small <- titanic[sample(1:nrow(titanic), 500), c(1,2,6,9)]
model_titanic_glm <- glm(survived == "yes" ~ gender + age + fare,
                        data = titanic_small, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_small[,-9],
                              y = titanic_small$survived == "yes")
bd_rf <- break_down_uncertainty(explain_titanic_glm, titanic_small[1, ])
bd_rf
plot(bd_rf)
```

Not run:

```
library("randomForest")
set.seed(1313)
model <- randomForest(status ~ . , data = HR)
new_observation <- HR_test[1,]

explainer_rf <- explain(model,
                       data = HR[1:1000,1:5],
                       y = HR$status[1:1000])
```

```
bd_rf <- break_down_uncertainty(explainer_rf,
                              new_observation)
bd_rf
```

```
# example for regression - apartment prices
# here we do not have interactions
model <- randomForest(m2.price ~ . , data = apartments)
explainer_rf <- explain(model,
                       data = apartments_test[1:1000,2:6],
                       y = apartments_test$m2.price[1:1000])
```

```
bd_rf <- break_down_uncertainty(explainer_rf, apartments_test[1,])
bd_rf
```

Index

`break_down`, [2](#), [5–7](#), [9](#), [12–14](#), [17](#), [21](#)
`break_down_uncertainty`, [4](#), [16](#), [22](#)

`describe`, [6](#)

`explain`, [3](#), [4](#), [9](#), [11](#)

`local_attributions`, [2](#), [3](#), [5](#), [8](#), [12](#), [14](#)
`local_interactions`, [2](#), [3](#), [9](#), [11](#), [14](#)

`plot.break_down`, [13](#)
`plot.break_down_uncertainty`, [15](#)
`plotD3`, [17](#)
`plotD3.shap`, [19](#)
`print.break_down`, [21](#)
`print.break_down_description`, [22](#)
`print.break_down_uncertainty`, [22](#)

`round`, [13](#), [18](#), [19](#), [21](#)

`shap`, [6](#), [7](#), [19](#)
`shap (break_down_uncertainty)`, [4](#)
`signif`, [13](#), [18](#), [19](#), [21](#)