

# Package: iBreakDown (via r-universe)

August 26, 2024

**Title** Model Agnostic Instance Level Variable Attributions

**Version** 2.1.2

**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](https://doi.org/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](https://arxiv.org/abs/1903.11420)> This package is a part of the 'DrWhy.AI' universe (Biecek 2018) <[arXiv:1806.08915](https://arxiv.org/abs/1806.08915)>.

**Depends** R (>= 3.5)

**License** GPL-3

**Encoding** UTF-8

**RoxygenNote** 7.2.2

**Imports** ggplot2

**Suggests** DALEX, knitr, rmarkdown, randomForest, e1071, ranger, nnet, testthat, r2d3, jsonlite, covr

**VignetteBuilder** knitr

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

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

**Repository** <https://modeloriented.r-universe.dev>

**RemoteUrl** <https://github.com/modeloriented/ibreakdown>

**RemoteRef** HEAD

**RemoteSha** d4f6289f9d9358a097b52918955ce55bd855a0c8

## Contents

break_down . . . . .	2
break_down_uncertainty . . . . .	4
describe . . . . .	7
local_attributions . . . . .	9
local_interactions . . . . .	11
plot.break_down . . . . .	14
plot.break_down_uncertainty . . . . .	17
plotD3 . . . . .	19
plotD3.shap . . . . .	21
print.break_down . . . . .	23
print.break_down_description . . . . .	24
print.break_down_uncertainty . . . . .	24

<b>Index</b>	<b>26</b>
--------------	-----------

---

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 <code>local_*</code> 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 <code>plot()</code> .
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**

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

**See Also**

[local\\_attributions](#), [local\\_interactions](#)

**Examples**

```
library("DALEX")
library("iBreakDown")
set.seed(1313)
model_titanic_glm <- glm(survived ~ gender + age + fare,
                        data = titanic_imputed, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_imputed,
                              y = titanic_imputed$survived,
                              label = "glm")

bd_glm <- break_down(explain_titanic_glm, titanic_imputed[1, ])
bd_glm
plot(bd_glm, max_features = 3)

## Not run:
## 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])

bd_rf <- break_down(explainer_rf,
                    new_observation)

head(bd_rf)
plot(bd_rf)

## End(Not run)

```

---

break\_down\_uncertainty

*Explanation Level Uncertainty of Sequential Variable Attribution*

---

## Description

This function calculates the break down algorithm for B random orderings. Then it calculates the 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],
  ...,

```



```

label = "glm")

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

## Not run:
## 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 <- break_down(explainer_rf,
                 apartments_test[1,])
plot(bd)

s <- shap(explainer_rf,
           apartments_test[1,])
plot(s)

## End(Not run)

```

---

`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_treshold = 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

<code>x</code>	an explanation created with <code>break_down</code> or <code>shap</code>
<code>nonsignificance_treshold</code>	a numeric specifying a threshold for variable importance
<code>...</code>	other arguments
<code>label</code>	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)
plot(bd_explanation)

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[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)

```

---

local\_attributions      *Model Agnostic Sequential Variable attributions*

---

## 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 `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 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

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

### See Also

[break\\_down](#), [local\\_interactions](#)

### Examples

```

library("DALEX")
library("iBreakDown")
set.seed(1313)
model_titanic_glm <- glm(survived ~ gender + age + fare,
                        data = titanic_imputed, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_imputed,
                              y = titanic_imputed$survived,
                              label = "glm")

bd_glm <- local_attributions(explain_titanic_glm, titanic_imputed[1, ])
bd_glm
plot(bd_glm, max_features = 3)

```

```

## Not run:
## 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])

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)

## End(Not run)

```

---

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()`. 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 <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 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.

**Value**

an object of the `break_down` class.

**References**

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

**See Also**

[break\\_down](#), [local\\_attributions](#)

**Examples**

```

library("DALEX")
library("iBreakDown")
set.seed(1313)
model_titanic_glm <- glm(survived ~ gender + age + fare,
                        data = titanic_imputed, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_imputed,
                              y = titanic_imputed$survived,
                              label = "glm")

bd_glm <- local_interactions(explain_titanic_glm, titanic_imputed[1, ],
                             interaction_preference = 500)
bd_glm
plot(bd_glm, max_features = 2)

## Not run:
library("randomForest")
# 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])

bd_rf <- local_interactions(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])

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)

```

```
## End(Not run)
```

---

plot.break_down	<i>Plot Generic for Break Down Objects</i>
-----------------	--

---

## 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,
  vnames = NULL,
  title = "Break Down profile",
  subtitle = "",
  max_vars = NULL
)
```

## Arguments

x	an explanation created with <a href="#">break_down</a>
...	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 10.
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	If NA (default), DrWhy colors are used.
digits	number of decimal places ( <a href="#">round</a> ) or significant digits ( <a href="#">signif</a> ) 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.
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 <code>keep_distributions=TRUE</code> in the <code>break_down</code> , <code>local_attributions</code> , or <code>local_interactions</code> .
vnames	a character vector, if specified then will be used as labels on OY axis. By default NULL
title	a character. Plot title. By default "Break Down profile".
subtitle	a character. Plot subtitle. By default "".
max_vars	alias for the <code>max_features</code> parameter.

**Value**

a ggplot2 object.

**References**

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

**Examples**

```
library("DALEX")
library("iBreakDown")
set.seed(1313)
model_titanic_glm <- glm(survived ~ gender + age + fare,
                        data = titanic_imputed, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                             data = titanic_imputed,
                             y = titanic_imputed$survived,
                             label = "glm")

bd_glm <- break_down(explain_titanic_glm, titanic_imputed[1, ])
bd_glm
plot(bd_glm, max_features = 3)
plot(bd_glm, max_features = 3,
     vnames = c("average", "+ male", "+ young", "+ cheap ticket", "+ other factors", "final"))

## Not run:
## Not run:
library("randomForest")
```





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

## End(Not run)
```

---

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,
  max_features = 10,
  max_vars = NULL
)
```

## Arguments

x	an explanation created with <a href="#">break_down_uncertainty</a>
...	other parameters.
vcolors	If NA (default), DrWhy colors are used.
show_boxplots	logical if TRUE (default) boxplot will be plotted to show uncertainty of attributions
max_features	maximal number of features to be included in the plot. By default it's 10.
max_vars	alias for the max_features parameter.

## Value

a ggplot2 object.

## References

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

**Examples**

```

library("DALEX")
library("iBreakDown")
set.seed(1313)
model_titanic_glm <- glm(survived ~ gender + age + fare,
                        data = titanic_imputed, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_imputed,
                              y = titanic_imputed$survived,
                              label = "glm")

sh_glm <- shap(explain_titanic_glm, titanic_imputed[1, ])

sh_glm
plot(sh_glm)

## Not run:
## 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,
                              path = c(3,2,4,1,5),
                              show_boxplots = FALSE)

bd_rf
plot(bd_rf, max_features = 3)

# 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)

## End(Not run)
```

---

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,
  max_vars = NULL,
  reload = FALSE
)
```

## Arguments

<code>x</code>	an explanation created with <code>break_down</code>
<code>...</code>	other parameters.
<code>baseline</code>	if numeric then vertical line will start in baseline.
<code>max_features</code>	maximal number of features to be included in the plot. By default it's 10.
<code>digits</code>	number of decimal places ( <code>round</code> ) or significant digits ( <code>signif</code> ) to be used. See the <code>rounding_function</code> argument.
<code>rounding_function</code>	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.

<code>bar_width</code>	width of bars in px. By default it's 12px
<code>margin</code>	extend x axis domain range to adjust the plot. Usually value between 0.1 and 0.3, by default it's 0.2
<code>scale_height</code>	if TRUE, the height of the plot scales with window size
<code>min_max</code>	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.
<code>vcolors</code>	If NA (default), DrWhy colors are used.
<code>chart_title</code>	a character. Set custom title
<code>time</code>	in ms. Set the animation length
<code>max_vars</code>	alias for the <code>max_features</code> parameter.
<code>reload</code>	Reload the plot on resize. By default it's FALSE.

### Value

a r2d3 object.

### References

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

### Examples

```
library("DALEX")
library("iBreakDown")
set.seed(1313)
model_titanic_glm <- glm(survived ~ gender + age + fare,
                        data = titanic_imputed, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_imputed,
                              y = titanic_imputed$survived,
                              label = "glm")

bd_glm <- local_attributions(explain_titanic_glm, titanic_imputed[1, ])
bd_glm
plotD3(bd_glm)

## Not run:
## Not run:
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)

## End(Not run)

```

---

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,
  max_vars = NULL,
  reload = FALSE
)

```

## Arguments

x	an explanation created with <a href="#">shap</a>
...	other parameters.
baseline	if numeric then vertical line will start in baseline.
max_features	maximal number of features to be included in the plot. By default it's 10.
digits	number of decimal places ( <a href="#">round</a> ) or significant digits ( <a href="#">signif</a> ) to be used. See the <code>rounding_function</code> 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 it's 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 the animation length
max_vars	alias for the <code>max_features</code> parameter.
reload	Reload the plot on resize. By default it's FALSE.

**Value**

a r2d3 object.

**References**

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

**Examples**

```
library("DALEX")
library("iBreakDown")
set.seed(1313)
model_titanic_glm <- glm(survived ~ gender + age + fare,
                        data = titanic_imputed, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                             data = titanic_imputed,
                             y = titanic_imputed$survived,
                             label = "glm")

s_glm <- shap(explain_titanic_glm, titanic_imputed[, ])
s_glm
plotD3(s_glm)

## Not run:
## Not run:
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)

## End(Not run)
```

---

print.break\_down      *Print Generic for Break Down Objects*

---

## 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

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

---

```
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 character

**References**

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

---

```
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**

Explanatory Model Analysis. Explore, Explain and Examine Predictive Models. <https://ema.drwhy.ai>

**Examples**

```
library("DALEX")
library("iBreakDown")
set.seed(1313)
model_titanic_glm <- glm(survived ~ gender + age + fare,
                        data = titanic_imputed, family = "binomial")
explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_imputed,
                              y = titanic_imputed$survived,
                              label = "glm")

bd_glm <- break_down_uncertainty(explain_titanic_glm, titanic_imputed[1, ])
bd_glm
plot(bd_glm)

## Not run:
## 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],
                       verbose = FALSE)

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

## End(Not run)
```

# Index

`break_down`, [2](#), [5](#), [7–10](#), [13–15](#), [19](#), [23](#)  
`break_down_uncertainty`, [4](#), [17](#), [24](#)

`describe`, [7](#)

`explain`, [3](#), [5](#), [10](#), [12](#)

`local_attributions`, [2](#), [3](#), [5](#), [9](#), [13](#), [15](#)  
`local_interactions`, [2](#), [3](#), [10](#), [11](#), [15](#)

`plot.break_down`, [14](#)  
`plot.break_down_uncertainty`, [17](#)  
`plotD3`, [19](#)  
`plotD3.shap`, [21](#)  
`print.break_down`, [23](#)  
`print.break_down_description`, [24](#)  
`print.break_down_uncertainty`, [24](#)

`round`, [14](#), [15](#), [19](#), [21–23](#)

`shap`, [7](#), [8](#), [21](#)  
`shap (break_down_uncertainty)`, [4](#)  
`signif`, [14](#), [15](#), [19](#), [21–23](#)