

Package ‘breakDown’

June 14, 2018

Title Model Agnostic Explainers for Individual Predictions

Version 0.1.6

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

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

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

This package work for binary classifiers and general regression models.

Depends R (>= 3.0)

Date 2018-06-11

License GPL-2

Encoding UTF-8

LazyData true

Imports ggplot2

RoxygenNote 6.0.1

Suggests knitr, rmarkdown, e1071, kernlab, xgboost, caret,
randomForest, ranger, testthat

VignetteBuilder knitr

URL <https://pbiecek.github.io/breakDown/>

BugReports <https://github.com/pbiecek/breakDown/issues>

NeedsCompilation no

Author Przemyslaw Biecek [aut, cre],
Aleksandra Grudziaz [ctb]

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

Repository CRAN

Date/Publication 2018-06-14 18:33:11 UTC

R topics documented:

betas	2
broken	2

broken.default	3
broken.glm	5
broken.lm	6
HR_data	7
plot.broken	8
print.broken	9
wine	10

Index	11
--------------	-----------

betas	<i>Extract betas values of a model for specific observations</i>
-------	--

Description

Extract betas values of a model for specific observations

Usage

```
betas(object, newdata, ...)
```

Arguments

object	a model
newdata	new observation(s) with columns that correspond to variables used in the model
...	unused additional parameters

Author(s)

Joseph Larmarange

broken	<i>Generic Function for Breaking Down of Model Predictions</i>
--------	--

Description

The broken function is a generic function for decomposition of model predictions. For linear models please use [broken.lm](#), for generic linear models please use [broken.glm](#). For all other models please use the model agnostic version [broken.default](#). Please note, that some of these functions have additional parameters.

Usage

```
broken(model, new_observation, ...)
```

Arguments

model a model
 new_observation a new observation with columns that corresponds to variables used in the model
 ... other parameters

Value

an object of the broken class

Examples

```

## Not run:
library("breakDown")
library("randomForest")
library("ggplot2")
set.seed(1313)
model <- randomForest(factor(left)~., data = HR_data, family = "binomial", maxnodes = 5)
predict.function <- function(model, new_observation)
  predict(model, new_observation, type="prob")[,2]
predict.function(model, HR_data[11,-7])
explain_1 <- broken(model, HR_data[11,-7], data = HR_data[, -7],
  predict.function = predict.function, direction = "down")
explain_1
plot(explain_1) + ggtitle("breakDown plot (direction=down) for randomForest model")

explain_2 <- broken(model, HR_data[11,-7], data = HR_data[, -7],
  predict.function = predict.function, direction = "down", keep_distributions = TRUE)
plot(explain_2, plot_distributions = TRUE) +
  ggtitle("breakDown distributions (direction=down) for randomForest model")

explain_3 <- broken(model, HR_data[11,-7], data = HR_data[, -7],
  predict.function = predict.function, direction = "up", keep_distributions = TRUE)
plot(explain_3, plot_distributions = TRUE) +
  ggtitle("breakDown distributions (direction=up) for randomForest model")

## End(Not run)

```

Description

This function implements two greedy strategies for decompositions of model predictions (see the direction parameter). Both strategies are model agnostic, they are greedy but in most cases they give very similar results. Find more information about these strategies in <https://arxiv.org/abs/1804.01955>.

Usage

```
## Default S3 method:
broken(model, new_observation, data, direction = "up", ...,
       baseline = 0, keep_distributions = FALSE, predict.function = predict)
```

Arguments

<code>model</code>	a model, it can be any predictive model, find examples for most popular frameworks in vignettes
<code>new_observation</code>	a new observation with columns that corresponds to variables used in the model
<code>data</code>	the original data used for model fitting, should have same columns as the 'new_observation'.
<code>direction</code>	either 'up' or 'down' determined the exploration strategy
<code>...</code>	other parameters
<code>baseline</code>	the origin/baseline for the breakDown plots, where the rectangles start. It may be a number or a character "Intercept". In the latter case the origin will be set to model intercept.
<code>keep_distributions</code>	if TRUE, then the distribution of partial predictions is stored in addition to the average.
<code>predict.function</code>	function that will calculate predictions out of model. It shall return a single numeric value per observation. For classification it may be a probability of the default class.

Value

an object of the broken class

Examples

```
## Not run:
library("breakDown")
library("randomForest")
library("ggplot2")
set.seed(1313)
model <- randomForest(factor(left)~., data = HR_data, family = "binomial", maxnodes = 5)
predict.function <- function(model, new_observation)
  predict(model, new_observation, type="prob")[,2]
predict.function(model, HR_data[11,-7])
explain_1 <- broken(model, HR_data[11,-7], data = HR_data[, -7],
  predict.function = predict.function, direction = "down")
explain_1
plot(explain_1) + ggtitle("breakDown plot (direction=down) for randomForest model")

explain_2 <- broken(model, HR_data[11,-7], data = HR_data[, -7],
  predict.function = predict.function, direction = "down", keep_distributions = TRUE)
plot(explain_2, plot_distributions = TRUE) +
```

```

ggtitle("breakDown distributions (direction=down) for randomForest model")

explain_3 <- broken(model, HR_data[11,-7], data = HR_data[,-7],
predict.function = predict.function, direction = "up", keep_distributions = TRUE)
plot(explain_3, plot_distributions = TRUE) +
  ggtitle("breakDown distributions (direction=up) for randomForest model")

## End(Not run)

```

broken.glm

Breaking Down of Model Predictions for glm models

Description

Breaking Down of Model Predictions for glm models

Usage

```

## S3 method for class 'glm'
broken(model, new_observation, ..., baseline = 0,
predict.function = stats::predict.glm)

```

Arguments

model	a glm model
new_observation	a new observation with columns that corresponds to variables used in the model
...	other parameters
baseline	the origin/baseline for the breakDown plots, where the rectangles start. It may be a number or a character "Intercept". In the latter case the origin will be set to model intercept.
predict.function	function that will calculate predictions out of model (typically predict or betas)

Value

an object of the broken class

Examples

```

# example for wine data
wine$qualityb <- factor(wine$quality > 5.5, labels = c("bad", "good"))
modelg <- glm(qualityb~fixed.acidity + volatile.acidity + citric.acid +
residual.sugar + chlorides + free.sulfur.dioxide +
total.sulfur.dioxide + density + pH + sulphates + alcohol,
data=wine, family = "binomial")
new_observation <- wine[1,]
br <- broken(modelg, new_observation)

```

```

logit <- function(x) exp(x)/(1+exp(x))
plot(br, logit)

# example for HR_data
model <- glm(left~., data = HR_data, family = "binomial")
explain_1 <- broken(model, HR_data[1,])
explain_1
plot(explain_1)
plot(explain_1, trans = function(x) exp(x)/(1+exp(x)))

explain_2 <- broken(model, HR_data[1,], predict.function = betas)
explain_2
plot(explain_2, trans = function(x) exp(x)/(1+exp(x)))

```

broken.lm

Breaking Down of Model Predictions for lm models

Description

Breaking Down of Model Predictions for lm models

Usage

```

## S3 method for class 'lm'
broken(model, new_observation, ..., baseline = 0,
       predict.function = stats::predict.lm)

```

Arguments

model	a lm model
new_observation	a new observation with columns that corresponds to variables used in the model
...	other parameters
baseline	the origin/baseline for the breakDown plots, where the rectangles start. It may be a number or a character "Intercept". In the latter case the origin will be set to model intercept.
predict.function	function that will calculate predictions out of model (typically predict or betas)

Value

an object of the broken class

Examples

```
model <- lm(Sepal.Length~., data=iris)
new_observation <- iris[1,]
br <- broken(model, new_observation)
plot(br)

# works for interactions as well
model <- lm(Sepal.Length ~ Petal.Width*Species, data = iris)
summary(model)

new_observation <- iris[1,]
br <- broken(model, new_observation)
br
plot(br)

br2 <- broken(model, new_observation, predict.function = betas)
br2
plot(br2)
```

HR_data

Why are our best and most experienced employees leaving prematurely?

Description

A dataset from Kaggle competition Human Resources Analytics. <https://www.kaggle.com/>

Format

A data frame with 14999 rows and 10 variables

Details

- satisfaction_level Level of satisfaction (0-1)
- last_evaluation Time since last performance evaluation (in Years)
- number_project Number of projects completed while at work
- average_monthly_hours Average monthly hours at workplace
- time_spend_company Number of years spent in the company
- Work_accident Whether the employee had a workplace accident
- left Whether the employee left the workplace or not (1 or 0) Factor
- promotion_last_5years Whether the employee was promoted in the last five years
- sales Department in which they work for
- salary Relative level of salary (high)

Source

Dataset HR-analytics from <https://www.kaggle.com>

 plot.broken

Break Down Plot

Description

Break Down Plot

Usage

```
## S3 method for class 'broken'
plot(x, trans = I, ..., top_features = 0, min_delta = 0,
     add_contributions = TRUE, vcolors = c(`-1` = "#d8b365", `0` = "#f5f5f5",
     `1` = "#5ab4ac", X = "darkgrey"), digits = 3, rounding_function = round,
     plot_distributions = FALSE)
```

Arguments

x	the model model of 'broken' class
trans	transformation that shall be applied to scores
...	other parameters
top_features	maximal number of variables from model we want to plot
min_delta	minimal stroke value of variables from model we want to plot
add_contributions	shall variable contributions to be added on plot?
vcolors	named vector with colors
digits	number of decimal places (round) or significant digits (signif) to be used. See the rounding_function argument
rounding_function	function that is to used for rounding numbers. It may be signif() which keeps a specified number of significant digits. Or the default round() to have the same precision for all components
plot_distributions	if TRUE then distributions of conditional proportions will be plotted. This requires keep_distributions=TRUE in the broken.default().

Value

a ggplot2 object

Examples

```
## Not run:
library("breakDown")
library("randomForest")
library("ggplot2")
```

```

set.seed(1313)
model <- randomForest(factor(left)~., data = HR_data, family = "binomial", maxnodes = 5)
predict.function <- function(model, new_observation)
  predict(model, new_observation, type="prob")[,2]
predict.function(model, HR_data[11,-7])
explain_1 <- broken(model, HR_data[11,-7], data = HR_data[, -7],
  predict.function = predict.function, direction = "down")
explain_1
plot(explain_1) + ggtitle("breakDown plot (direction=down) for randomForest model")

explain_2 <- broken(model, HR_data[11,-7], data = HR_data[, -7],
  predict.function = predict.function, direction = "down", keep_distributions = TRUE)
plot(explain_2, plot_distributions = TRUE) +
  ggtitle("breakDown distributions (direction=down) for randomForest model")

explain_3 <- broken(model, HR_data[11,-7], data = HR_data[, -7],
  predict.function = predict.function, direction = "up", keep_distributions = TRUE)
plot(explain_3, plot_distributions = TRUE) +
  ggtitle("breakDown distributions (direction=up) for randomForest model")

model <- lm(quality~., data=wine)
new_observation <- wine[1,]
br <- broken(model, new_observation)
plot(br)
plot(br, top_features = 2)
plot(br, top_features = 2, min_delta = 0.01)

## End(Not run)

```

print.broken

Break Down Print

Description

Break Down Print

Usage

```

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

```

Arguments

x	the model model of 'broken' class
...	other parameters
digits	number of decimal places (round) or significant digits (signif) to be used. See the rounding_function argument

rounding_function

function that is to be used for rounding numbers. It may be `signif()` which keeps a specified number of significant digits. Or the default `round()` to have the same precision for all components

Value

a data frame

wine

White Wine Quality Data

Description

White wine quality data related to variants of the Portuguese "Vinho Verde" wine. For more details, consult: <http://www.vinhoverde.pt/en/> or the reference Cortez et al., 2009.

Format

A data frame with 4898 rows and 12 variables

Details

A dataset downloaded from UCI Machine Learning Database <https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.cs>

- fixed.acidity
- volatile.acidity
- citric.acid
- residual.sugar
- chlorides
- free.sulfur.dioxide
- total.sulfur.dioxide
- density
- pH
- sulphates
- alcohol
- quality

Source

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In *Decision Support Systems*, Elsevier, 47(4):547-553. ISSN: 0167-9236.

Index

*Topic **datasets**

HR_data, [7](#)

betas, [2](#)

broken, [2](#)

broken.default, [2](#), [3](#)

broken.glm, [2](#), [5](#)

broken.lm, [2](#), [6](#)

HR_data, [7](#)

plot.broken, [8](#)

print.broken, [9](#)

wine, [10](#)