Package 'modelStudio'

January 21, 2020

Title Interactive Studio with Explanations for ML Predictive Models

Version 0.2.1

Description Automate explanation of machine learning predictive models.
This package generates advanced interactive and animated model explanations in the form of a serverless HTML site. It combines 'R' with 'D3.js' to produce plots and descriptions for various local and global explanations. Tools for model exploration unite with tools for EDA (Exploratory Data Analysis) to give a broad overview of the model behaviour. 'modelStudio' is a fast and condensed way to get all the answers without much effort. Break down your model and look into its ingredients with only a few lines of code.

Depends R (>= 3.5)

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Imports iBreakDown (>= 0.9.9), ingredients (>= 0.5), r2d3, jsonlite

Suggests DALEX (>= 0.4.9), parallelMap, randomForest, knitr, rmarkdown, testthat, spelling

VignetteBuilder knitr

URL https://ModelOriented.github.io/modelStudio/, https://github.com/ModelOriented/modelStudio

BugReports https://github.com/ModelOriented/modelStudio/issues

Language en-US

NeedsCompilation no

Author Hubert Baniecki [aut, cre] (<https://orcid.org/0000-0001-6661-5364>), Przemyslaw Biecek [aut] (<https://orcid.org/0000-0001-8423-1823>)

Maintainer Hubert Baniecki <hbaniecki@gmail.com>

Repository CRAN

Date/Publication 2020-01-21 11:50:02 UTC

R topics documented:

`modelStudio` *Generate Interactive Studio with Explanations for the Model*

Description

This tool uses your model, data and new observations, to provide local and global explanations. It generates plots and descriptions in the form of the serverless HTML site, that supports animations and interactivity made with D3.js.

Find more details about plots in Explanatory Model Analysis: Explore, Explain and Examine Predictive Models

Usage

```
modelStudio(object, ...)

## S3 method for class 'explainer'
modelStudio(
  object,
  new_observation = NULL,
  new_observation_y = NULL,
  facet_dim = c(2, 2),
  time = 500,
  max_features = 10,
  N = 400,
  B = 15,
  show_info = TRUE,
  parallel = FALSE,
  options = modelStudioOptions(),
  viewer = "external",
  ...
)

## Default S3 method:
modelStudio(
  object,
  data,
  y,
  predict_function = predict,
  label = class(model)[1],
  new_observation = NULL,
  new_observation_y = NULL,
```

```
  facet_dim = c(2, 2),
  time = 500,
  max_features = 10,
  N = 400,
  B = 15,
  show_info = TRUE,
  parallel = FALSE,
  options = modelStudioOptions(),
  viewer = "external",
  ...
)
```

Arguments

`object`	An `explainer` created with function `DALEX::explain()` or a model to be explained.
`...`	Other parameters.
`new_observation`	A new observation with columns that correspond to variables used in the model.
`new_observation_y`	True label for `new_observation`.
`facet_dim`	Dimensions of the grid. Default is `c(2,2)`.
`time`	Time in ms. Set animation length. Default is `500`.
`max_features`	Maximum number of features to be included in Break Down and SHAP Values plots. Default is `10`.
`N`	Number of observations used for calculation of partial dependency profiles. Default is `400`.
`B`	Number of random paths used for calculation of SHAP values. Default is `15`.
`show_info`	Verbose progress bar on the console. Default is `TRUE`.
`parallel`	Speed up the computation using `parallelMap::parallelMap()`. See **vignette**.
`options`	Customize `modelStudio`. See `modelStudioOptions` and **vignette**.
`viewer`	Default is `external` to display in an external RStudio window. Use `browser` to display in an external browser or `internal` to use the RStudio internal viewer pane for output.
`data`	Validation dataset, will be extracted from `object` if it is an explainer. NOTE: It is best when target variable is not present in the `data`.
`y`	True labels for `data`, will be extracted from `object` if it is an `explainer`.
`predict_function`	Predict function, will be extracted from `object` if it is an `explainer`.
`label`	A name of the model, will be extracted from `object` if it is an `explainer`.

Value

An object of the `r2d3` class.

References

- Wrapper for the function is implemented in **DALEX**
- Feature Importance, Ceteris Paribus, Partial Dependency and Accumulated Dependency plots are implemented in **ingredients**
- Break Down and SHAP Values plots are implemented in **iBreakDown**

See Also

Python wrappers and more can be found in **DALEXtra**

Examples

```
library("modelStudio")

# ex1 classification

model_titanic_glm <- glm(survived ~.,
                         data = DALEX::titanic_imputed,
                         family = "binomial")

explain_titanic_glm <- DALEX::explain(model_titanic_glm,
                                      data = DALEX::titanic_imputed[,-8],
                                      y = DALEX::titanic_imputed[,8],
                                      label = "glm",
                                      verbose = FALSE)

new_observations <- DALEX::titanic_imputed[1:2,]
rownames(new_observations) <- c("Lucas","James")

modelStudio(explain_titanic_glm, new_observations,
            N = 100, B = 10, show_info = FALSE)


# ex2 regression

model_apartments <- glm(m2.price ~. ,
                        data = DALEX::apartments)

explain_apartments <- DALEX::explain(model_apartments,
                                     data = DALEX::apartments[,-1],
                                     y = DALEX::apartments[,1],
                                     verbose = FALSE)

new_apartments <- DALEX::apartments[1:2,]
rownames(new_apartments) <- c("ap1","ap2")

modelStudio(explain_apartments, new_apartments,
            facet_dim = c(2, 3), time = 1000,
            show_info = FALSE)

modelStudio(explain_apartments, show_info = FALSE)
```

```
modelStudio(explain_apartments, new_observation = new_apartments,
                                new_observation_y = DALEX::apartments[1:2, 1],
                                show_info = FALSE)
```

`modelStudioOptions` *Modify options and pass them to modelStudio*

Description

This function returns default options for `modelStudio`. It is possible to modify values of this list and pass it to the `options` parameter in the main function. **WARNING: Editing default options may cause unintended behavior.**

Usage

```
modelStudioOptions(...)
```

Arguments

`...`	Options to change, `option_name = value`.

Value

`list` of options for `modelStudio`.

Main options::

scale_plot `TRUE` Makes every plot the same height, ignores `bar_width`.
show_subtitle `TRUE` Should the subtitle be displayed?
subtitle `label` parameter from `explainer`.
margin_* Plot margins. Change `margin_left` for longer/shorter axis labels.
w `420` in px. Inner plot width.
h `280` in px. Inner plot height.
bar_width `16` in px. Default width of bars for all plots, ignored when `scale_plot = TRUE`.
line_size `2` in px. Default width of lines for all plots.
point_size `3` in px. Default point radius for all plots.
[bar,line,point _color] `[#46bac2,#46bac2,#371ea3]`
positive_color `#8bdcbe` for Break Down and SHAP Values bars.
negative_color `#f05a71` for Break Down and SHAP Values bars.
default_color `#371ea3` for Break Down bar and highlighted line.

Plot specific options:: `**` is a two letter code unique to each plot, might be one of `[bd,sv,cp,fi,pd,ad,fd,tv,at]`.

****_title** Plot specific title. Default varies.

**_subtitle Plot specific subtitle. Default is subtitle.

**_bar_width Plot specific width of bars. Default is bar_width, ignored when scale_plot = TRUE.

**_line_size line_size Plot specific width of lines. Default is line_size.

**_point_size Plot specific point radius. Default is point_size.

**_*_color Plot specific [bar,line,point] color. Default is [bar,line,point]_color.

Examples

```
library("modelStudio")

apartments <- DALEX::apartments

model_apartments <- glm(m2.price ~. ,
                        data = apartments)

explain_apartments <- DALEX::explain(model_apartments,
                                     data = apartments[,-1],
                                     y = apartments[,1],
                                     verbose = FALSE)

new_apartments <- apartments[1:2,]
rownames(new_apartments) <- c("ap1","ap2")

op <- modelStudioOptions(
  show_subtitle = TRUE,
  bd_subtitle = "Hello World",
  line_size = 5,
  point_size = 9,
  line_color = "pink",
  point_color = "purple",
  bd_positive_color = "yellow",
  bd_negative_color = "orange"
)

modelStudio(explain_apartments, new_apartments,
            facet_dim = c(1,2), N = 100, B = 10, show_info = FALSE,
            options = op)
```

Index