triplot: : SUMMARY

Overview

Tools for exploration and explanation of machine learning models.

triplot explains the attributions of correlated features to model's prediction.

- Delivers an instance-level explainer predict_aspects() that supports calculating the importance of the groups of explanatory variables.
- Provides a tool called triplot that shows instance- and data-level summary of automatic aspect importance grouping,

The triplot package is a part of DrWhy.AI universe. More information about analysis of machine learning models can be found in the Explanatory Model Analysis. Explore, Explain and Examine Predictive Models e-book.

using predict_aspects with lasso

Predict_aspects() can calculate aspects' importance by using either linear regression or **lasso regression**. Using lasso, we can control how many nonzero aspects importance values are present in the final explanation. To use lasso, n_var parameter has to be provided. It declares **how many aspects importance nonzero values** we would like to get.

predict_aspects – list of additional parameters

- n_var - maximum number of non-zero coefficients after **lasso fitting** (if zero, than linear regression is used)
- sample_method - sampling method in get_sample()
- f - frequency in get_sample()

group_variables function

Divides correlated features into groups, called aspects. Division is based on correlation cut-off level (features min. pairwise correlation in one group in at least at level h).

```
group_variables(apartments_num, h = 0.5)
```

DOCUMENTATION

- webpage
- overview (vignette)
- FIFA usecase (vignette)
- method description (vignette)
- Source code for this cheatsheet

predict_aspects

predict_aspects() allows to **calculate contribution** to the prediction of the groups of explanatory variables (called aspects)

INTUITION

Function uses subset of observations from the original dataset and than it modifies it, so every observation will have at least one aspect replaced by the data from the observation of interest. Then it builds model that predicts how those replacements change predictions.

BASIC EXAMPLE

predict_aspects() works on DALEX explainers.

```
model_ap <- lm(m2.price ~ ., data = apartments_num)
explain <- explain(model_ap, data = apartments_num)
```

After creating an explainer, we are manually choosing variables into aspects.

```
aspects <- list(
  living.area = c("surface", "no.rooms"),
  construction.year = "construction.year",
  floor = "floor")
```

We are defining a new observation for which we will be explaining the prediction.

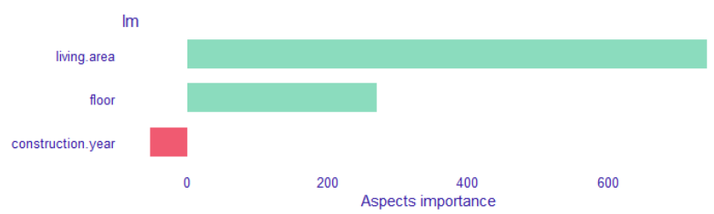
```
new_apartment <- data.frame(construction.year =
1985, surface = 25, floor = 3, no.rooms = 1)
```

Importance is calculated by predict_aspects() function. We can check the results with generics **print** and **plot**.

```
pa <- predict_aspects(
  x = explain,
  new_observation = new_apartment,
  variable_groups = aspects)

print(pa, show_features = TRUE)
##     variable_groups importance          features
## 2       living.area     727.33 surface, no.rooms
## 4              floor     219.40             floor
## 3 construction.year     -60.22 construction.year

plot(pa)
```



triplot

triplot shows, in one place:

- the importance of every **single feature**,
- **hierarchical** aspects importance,
- **order** of grouping features into aspects.

We can use triplot to investigate the **instance level importance** of features or to illustrate the **model level importance** of features. triplot can be only used on numerical features.

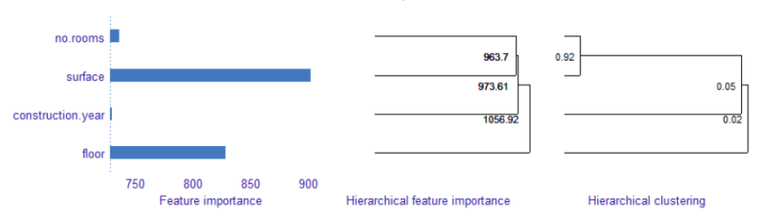
triplot works on DALEX explainers.

```
explain <- explain(model_ap,
                   data = apartments_no_target,
                   y = apartments_num$m2.price)
```

model_triplot

With model_triplot() we calculate the triplot object and then plot it with the generic plot() function.

```
triplot <- model_triplot(explain)
plot(triplot)
```



predict_triplot

To investigate **instance level** feature importance we use predict_triplot() and plot() functions.

```
triplot <- predict_triplot(explain,
  new_observation = new_apartment)
plot(triplot)
```