

Package ‘triplot’

July 13, 2020

Title Explaining Correlated Features in Machine Learning Models

Version 1.3.0

Description Tools for exploring effects of correlated features in predictive models. The `predict_tripplot()` function delivers instance-level explanations that calculate the importance of the groups of explanatory variables. The `model_tripplot()` function delivers data-level explanations. The generic plot function visualises in a concise way importance of hierarchical groups of predictors. All of the tools are model agnostic, therefore works for any predictive machine learning models. Find more details in Biecek (2018) <arXiv:1806.08915>.

Depends R (>= 3.6)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports ggplot2, DALEX (>= 1.3), glmnet, ggdendro, patchwork

Suggests testthat, knitr, randomForest, mlbench, ranger, gbm, covr

URL <https://github.com/ModelOriented/triplot>

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

Language en-US

NeedsCompilation no

Author Katarzyna Pekala [aut, cre],
Przemyslaw Biecek [aut] (<<https://orcid.org/0000-0001-8423-1823>>)

Maintainer Katarzyna Pekala <katarzyna.pekala@gmail.com>

Repository CRAN

Date/Publication 2020-07-13 17:00:03 UTC

R topics documented:

| | |
|------------------------------------|----|
| aspect_importance | 2 |
| aspect_importance_single | 4 |
| calculate_triplet | 6 |
| cluster_variables | 8 |
| get_sample | 9 |
| group_variables | 10 |
| hierarchical_importance | 10 |
| list_variables | 12 |
| plot.aspect_importance | 13 |
| plot.cluster_variables | 14 |
| plot.triplet | 15 |
| print.aspect_importance | 16 |

| | |
|--------------|-----------|
| Index | 18 |
|--------------|-----------|

| | |
|-------------------|---|
| aspect_importance | <i>Calculates importance of variable groups (called aspects) for a selected observation</i> |
|-------------------|---|

Description

Predict aspects function takes a sample from a given dataset and modifies it. Modification is made by replacing part of its aspects by values from the observation. Then function is calculating the difference between the prediction made on modified sample and the original sample. Finally, it measures the impact of aspects on the change of prediction by using the linear model or lasso.

Usage

```
aspect_importance(x, ...)

## S3 method for class 'explainer'
aspect_importance(
  x,
  new_observation,
  variable_groups,
  N = 1000,
  n_var = 0,
  sample_method = "default",
  f = 2,
  ...
)

## Default S3 method:
aspect_importance(
  x,
  data,
```

```

    predict_function = predict,
    label = class(x)[1],
    new_observation,
    variable_groups,
    N = 100,
    n_var = 0,
    sample_method = "default",
    f = 2,
    ...
)

lime(x, ...)

predict_aspects(x, ...)

```

Arguments

| | |
|------------------|--|
| x | an explainer created with the <code>DALEX::explain()</code> function or a model to be explained. |
| ... | other parameters |
| new_observation | selected observation with columns that corresponds to variables used in the model |
| variable_groups | list containing grouping of features into aspects |
| N | number of observations to be sampled (with replacement) from data NOTE: Small N may cause unstable results. |
| 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 |
| data | dataset, it will be extracted from x if it's an explainer NOTE: It is best when target variable is not present in the data |
| predict_function | predict function, it will be extracted from x if it's an explainer |
| label | name of the model. By default it's extracted from the 'class' attribute of the model. |

Value

An object of the class `aspect_importance`. Contains data frame that describes aspects' importance.

Examples

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

```

model_titanic_glm <- glm(survived == 1 ~
                        class+gender+age+sibsp+parch+fare+embarked,
                        data = titanic_imputed,
                        family = "binomial")

explain_titanic_glm <- explain(model_titanic_glm,
                              data = titanic_imputed[,-8],
                              y = titanic_imputed$survived == 1,
                              verbose = FALSE)

aspects <- list(wealth = c("class", "fare"),
               family = c("sibsp", "parch"),
               personal = c("gender", "age"),
               embarked = "embarked")

predict_aspects(explain_titanic_glm,
                new_observation = titanic_imputed[1,],
                variable_groups = aspects)

library("randomForest")
library("DALEX")
model_titanic_rf <-
  randomForest(factor(survived) ~ class + gender + age + sibsp +
              parch + fare + embarked,
              data = titanic_imputed)

explain_titanic_rf <- explain(model_titanic_rf,
                             data = titanic_imputed[,-8],
                             y = titanic_imputed$survived == 1,
                             verbose = FALSE)

predict_aspects(explain_titanic_rf,
                new_observation = titanic_imputed[1,],
                variable_groups = aspects)

```

aspect_importance_single

Aspects importance for single aspects

Description

Calculates aspect_importance for single aspects (every aspect contains only one feature).

Usage

```
aspect_importance_single(x, ...)
```

```

## S3 method for class 'explainer'
aspect_importance_single(
  x,
  new_observation,
  N = 1000,
  n_var = 0,
  sample_method = "default",
  f = 2,
  ...
)

## Default S3 method:
aspect_importance_single(
  x,
  data,
  predict_function = predict,
  label = class(x)[1],
  new_observation,
  N = 1000,
  n_var = 0,
  sample_method = "default",
  f = 2,
  ...
)

```

Arguments

| | |
|------------------|--|
| x | an explainer created with the <code>DALEX::explain()</code> function or a model to be explained. |
| ... | other parameters |
| new_observation | selected observation with columns that corresponds to variables used in the model, should be without target variable |
| N | number of observations to be sampled (with replacement) from data NOTE: Small N may cause unstable results. |
| n_var | how many non-zero coefficients for lasso fitting, if zero than linear regression is used |
| sample_method | sampling method in get_sample |
| f | frequency in get_sample |
| data | dataset, it will be extracted from x if it's an explainer NOTE: Target variable shouldn't be present in the data |
| predict_function | predict function, it will be extracted from x if it's an explainer |
| label | name of the model. By default it's extracted from the 'class' attribute of the model. |

Value

An object of the class 'aspect_importance'. Contains dataframe that describes aspects' importance.

Examples

```
library("DALEX")
model_titanic_glm <- glm(survived == 1 ~ class + gender + age +
  sibsp + parch + fare + embarked,
  data = titanic_imputed,
  family = "binomial")

explainer_titanic <- explain(model_titanic_glm,
  data = titanic_imputed[, -8],
  verbose = FALSE)
aspect_importance_single(explainer_titanic,
  new_observation = titanic_imputed[1, -8])
```

| | |
|-------------------|--|
| calculate_triplet | <i>Calculate triplot that sums up automatic aspect/feature importance grouping</i> |
|-------------------|--|

Description

This function shows:

- plot for the importance of single variables,
- tree that shows importance for every newly expanded group of variables,
- clustering tree.

Usage

```
calculate_triplet(x, ...)

## S3 method for class 'explainer'
calculate_triplet(
  x,
  type = c("predict", "model"),
  new_observation = NULL,
  N = 1000,
  loss_function = DALEX::loss_root_mean_square,
  B = 10,
  fi_type = c("raw", "ratio", "difference"),
  clust_method = "complete",
  cor_method = "spearman",
  ...
)
```

```

## Default S3 method:
calculate_triplot(
  x,
  data,
  y = NULL,
  predict_function = predict,
  label = class(x)[1],
  type = c("predict", "model"),
  new_observation = NULL,
  N = 1000,
  loss_function = DALEX::loss_root_mean_square,
  B = 10,
  fi_type = c("raw", "ratio", "difference"),
  clust_method = "complete",
  cor_method = "spearman",
  ...
)

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

model_triplot(x, ...)

predict_triplot(x, ...)

```

Arguments

| | |
|-----------------|--|
| x | an explainer created with the <code>DALEX::explain()</code> function or a model to be explained. |
| ... | other parameters |
| type | if <code>predict</code> then <code>aspect_importance</code> is used, if <code>model</code> then <code>feature_importance</code> is calculated |
| new_observation | selected observation with columns that corresponds to variables used in the model, should be without target variable |
| N | number of rows to be sampled from data NOTE: Small N may cause unstable results. |
| loss_function | a function that will be used to assess variable importance, if <code>type = model</code> |
| B | integer, number of permutation rounds to perform on each variable in feature importance calculation, if <code>type = model</code> |
| fi_type | character, type of transformation that should be applied for dropout loss, if <code>type = model</code> . "raw" results raw drop losses, "ratio" returns <code>drop_loss/drop_loss_full_model</code> . |
| clust_method | the agglomeration method to be used, see hclust methods |
| cor_method | the correlation method to be used see cor methods |
| data | dataset, it will be extracted from x if it's an explainer NOTE: Target variable shouldn't be present in the data |

y true labels for data, will be extracted from x if it's an explainer
predict_function predict function, it will be extracted from x if it's an explainer
label name of the model. By default it's extracted from the 'class' attribute of the model.

Value

triplot object

Examples

```

library(DALEX)
set.seed(123)
apartments_num <- apartments[,unlist(lapply(apartments, is.numeric))]
apartments_num_lm_model <- lm(m2.price ~ ., data = apartments_num)
apartments_num_new_observation <- apartments_num[30, ]
explainer_apartments <- explain(model = apartments_num_lm_model,
                               data = apartments_num[,-1],
                               y = apartments_num[, 1],
                               verbose = FALSE)
apartments_tri <- calculate_triplot(x = explainer_apartments,
                                   new_observation =
                                   apartments_num_new_observation[-1])

apartments_tri
  
```

cluster_variables *Creates a cluster tree from numeric features*

Description

Creates a cluster tree from numeric features and their correlations.

Usage

```

cluster_variables(x, ...)

## Default S3 method:
cluster_variables(x, clust_method = "complete", cor_method = "spearman", ...)
  
```

Arguments

x dataframe with only numeric columns
... other parameters
clust_method the agglomeration method to be used see [hclust](#) methods
cor_method the correlation method to be used see [cor](#) methods

Value

an hclust object

Examples

```
library("DALEX")
dragons_data <- dragons[,c(2,3,4,7,8)]
cluster_variables(dragons_data, clust_method = "complete")
```

get_sample

Function for getting binary matrix

Description

Function creates binary matrix, to be used in aspect_importance method. It starts with a zero matrix. Then it replaces some zeros with ones. If sample_method = "default" it randomly replaces one or two zeros per row. If sample_method = "binom" it replaces random number of zeros per row - average number of replaced zeros can be controlled by parameter sample_method = "f". Function doesn't allow the returned matrix to have rows with only zeros.

Usage

```
get_sample(n, p, sample_method = c("default", "binom"), f = 2)
```

Arguments

| | |
|---------------|---------------------------------|
| n | number of rows |
| p | number of columns |
| sample_method | sampling method |
| f | frequency for binomial sampling |

Value

a binary matrix

Examples

```
get_sample(100,6,"binom",3)
```

| | |
|-----------------|--|
| group_variables | <i>Helper function that combines clustering variables and creating aspect list</i> |
|-----------------|--|

Description

Divides correlated features into groups, called aspects. Division is based on correlation cutoff level.

Usage

```
group_variables(x, h, clust_method = "complete", cor_method = "spearman")
```

Arguments

| | |
|--------------|--|
| x | hclust object |
| h | correlation value for tree cutting |
| clust_method | the agglomeration method to be used see hclust methods |
| cor_method | the correlation method to be used see cor methods |

Value

list with aspect

Examples

```
library("DALEX")
dragons_data <- dragons[,c(2,3,4,7,8)]
group_variables(dragons_data, h = 0.5, clust_method = "complete")
```

| | |
|-------------------------|--|
| hierarchical_importance | <i>Calculates importance of hierarchically grouped aspects</i> |
|-------------------------|--|

Description

This function creates a tree that shows order of feature grouping and calculates importance of every newly created aspect.

Usage

```

hierarchical_importance(
  x,
  data,
  y = NULL,
  predict_function = predict,
  type = "predict",
  new_observation = NULL,
  N = 1000,
  loss_function = DALEX::loss_root_mean_square,
  B = 10,
  fi_type = c("raw", "ratio", "difference"),
  clust_method = "complete",
  cor_method = "spearman",
  ...
)

## S3 method for class 'hierarchical_importance'
plot(
  x,
  absolute_value = FALSE,
  show_labels = TRUE,
  add_last_group = TRUE,
  axis_lab_size = 10,
  text_size = 3,
  ...
)

```

Arguments

| | |
|-------------------------------|--|
| <code>x</code> | a model to be explained. |
| <code>data</code> | dataset NOTE: Target variable shouldn't be present in the data |
| <code>y</code> | true labels for data |
| <code>predict_function</code> | predict function |
| <code>type</code> | if predict then aspect_importance is used, if model then feature_importance is calculated |
| <code>new_observation</code> | selected observation with columns that corresponds to variables used in the model, should be without target variable |
| <code>N</code> | number of rows to be sampled from data NOTE: Small N may cause unstable results. |
| <code>loss_function</code> | a function that will be used to assess variable importance, if type = model |
| <code>B</code> | integer, number of permutation rounds to perform on each variable in feature importance calculation, if type = model |

| | |
|----------------|--|
| fi_type | character, type of transformation that should be applied for dropout loss, if type = model. "raw" results raw drop losses, "ratio" returns drop_loss/drop_loss_full_model. |
| clust_method | the agglomeration method to be used, see hclust methods |
| cor_method | the correlation method to be used see cor methods |
| ... | other parameters |
| absolute_value | if TRUE, aspects importance values will be drawn as absolute values |
| show_labels | if TRUE, plot will have annotated axis Y |
| add_last_group | if TRUE, plot will draw connecting line between last two groups |
| axis_lab_size | size of labels on axis Y, if applicable |
| text_size | size of labels annotating values of aspects importance |

Value

ggplot

Examples

```
library(DALEX)
apartments_num <- apartments[,unlist(lapply(apartments, is.numeric))]
apartments_num_lm_model <- lm(m2.price ~ ., data = apartments_num)
hi <- hierarchical_importance(x = apartments_num_lm_model,
  data = apartments_num[, -1],
  y = apartments_num[, 1],
  type = "model")
plot(hi, add_last_group = TRUE, absolute_value = TRUE)
```

| | |
|----------------|--|
| list_variables | <i>Cuts tree at custom height and returns a list</i> |
|----------------|--|

Description

This function creates aspect list after cutting a cluster tree of features at a given height.

Usage

```
list_variables(x, h)
```

Arguments

| | |
|---|------------------------------------|
| x | hclust object |
| h | correlation value for tree cutting |

Value

list of aspects

Examples

```
library("DALEX")
dragons_data <- dragons[,c(2,3,4,7,8)]
cv <- cluster_variables(dragons_data, clust_method = "complete")
list_variables(cv, h = 0.5)
```

plot.aspect_importance

Function for plotting aspect_importance results

Description

This function plots the results of aspect_importance.

Usage

```
## S3 method for class 'aspect_importance'
plot(
  x,
  ...,
  bar_width = 10,
  show_features = aspects_on_axis,
  aspects_on_axis = TRUE,
  add_importance = FALSE,
  digits_to_round = 2,
  text_size = 3
)
```

Arguments

| | |
|-----------------|--|
| x | object of aspect_importance class |
| ... | other parameters |
| bar_width | bar width |
| show_features | if TRUE, labels on axis Y show aspect names, otherwise they show features names |
| aspects_on_axis | alias for show_features held for backwards compatibility |
| add_importance | if TRUE, plot is annotated with values of aspects importance |
| digits_to_round | integer indicating the number of decimal places used for rounding values of aspects importance shown on the plot |
| text_size | size of labels annotating values of aspects importance, if applicable |

Value

a ggplot2 object

Examples

```
library("DALEX")

model_titanic_glm <- glm(survived == 1 ~
  class+gender+age+sibsp+parch+fare+embarked,
  data = titanic_imputed,
  family = "binomial")

explain_titanic_glm <- explain(model_titanic_glm,
  data = titanic_imputed[,-8],
  y = titanic_imputed$survived == 1,
  verbose = FALSE)

aspects <- list(wealth = c("class", "fare"),
  family = c("sibsp", "parch"),
  personal = c("gender", "age"),
  embarked = "embarked")

titanic_ai <- predict_aspects(explain_titanic_glm,
  new_observation = titanic_imputed[1,],
  variable_groups = aspects)

plot(titanic_ai)
```

plot.cluster_variables

Plots tree with correlation values

Description

Plots tree that illustrates the results of cluster_variables function.

Usage

```
## S3 method for class 'cluster_variables'
plot(x, p = NULL, show_labels = TRUE, axis_lab_size = 10, text_size = 3, ...)
```

Arguments

| | |
|---------------|--|
| x | cluster_variables or hclust object |
| p | correlation value for cutoff level, if not NULL, cutoff line will be drawn |
| show_labels | if TRUE, plot will have annotated axis Y |
| axis_lab_size | size of labels on axis Y, if applicable |
| text_size | size of labels annotating values of correlations |
| ... | other parameters |

Value

plot

Examples

```
library("DALEX")
dragons_data <- dragons[,c(2,3,4,7,8)]
cv <- cluster_variables(dragons_data, clust_method = "complete")
plot(cv, p = 0.7)
```

plot.triplot

Plots triplot

Description

Plots triplot that sum up automatic aspect/feature importance grouping

Usage

```
## S3 method for class 'triplot'
plot(
  x,
  absolute_value = FALSE,
  add_importance_labels = FALSE,
  show_model_label = FALSE,
  abbrev_labels = 0,
  add_last_group = TRUE,
  axis_lab_size = 10,
  text_size = 3,
  bar_width = 5,
  margin_mid = 0.3,
  ...
)
```

Arguments

x triplot object

absolute_value if TRUE, aspect importance values will be drawn as absolute values

add_importance_labels if TRUE, first plot is annotated with values of aspects importance on the bars

show_model_label if TRUE, adds subtitle with model label

abbrev_labels if greater than 0, labels for axis Y in single aspect importance plot will be abbreviated according to this parameter

| | |
|----------------|--|
| add_last_group | if TRUE and type = predict, plot will draw connecting line between last two groups at the level of 105 biggest importance value, for model this line is always drawn at the baseline value |
| axis_lab_size | size of labels on axis |
| text_size | size of labels annotating values of aspects importance and correlations |
| bar_width | bar width in the first plot |
| margin_mid | size of a right margin of a middle plot |
| ... | other parameters |

Value

plot

Examples

```
library(DALEX)
set.seed(123)
apartments_num <- apartments[,unlist(lapply(apartments, is.numeric))]
apartments_num_lm_model <- lm(m2.price ~ ., data = apartments_num)
apartments_num_new_observation <- apartments_num[30, ]
explainer_apartments <- explain(model = apartments_num_lm_model,
                               data = apartments_num[,-1],
                               y = apartments_num[, 1],
                               verbose = FALSE)
apartments_tri <- calculate_triplet(x = explainer_apartments,
                                   new_observation = apartments_num_new_observation[-1])
plot(apartments_tri)
```

```
print.aspect_importance
```

Function for printing aspect_importance results

Description

This function prints the results of aspect_importance.

Usage

```
## S3 method for class 'aspect_importance'
print(x, show_features = FALSE, show_corr = FALSE, ...)
```

Arguments

| | |
|---------------|---|
| x | object of aspect_importance class |
| show_features | show list of features for every aspect |
| show_corr | show if all features in aspect are pairwise positively correlated (for numeric features only) |
| ... | other parameters |

Examples

```
library("DALEX")

model_titanic_glm <- glm(survived == 1 ~
  class+gender+age+sibsp+parch+fare+embarked,
  data = titanic_imputed,
  family = "binomial")

explain_titanic_glm <- explain(model_titanic_glm,
  data = titanic_imputed[,-8],
  y = titanic_imputed$survived == 1,
  verbose = FALSE)

aspects <- list(wealth = c("class", "fare"),
  family = c("sibsp", "parch"),
  personal = c("gender", "age"),
  embarked = "embarked")

titanic_ai <- predict_aspects(explain_titanic_glm,
  new_observation = titanic_imputed[1,],
  variable_groups = aspects)

print(titanic_ai)
```

Index

aspect_importance, 2
aspect_importance_single, 4

calculate_triplet, 6
cluster_variables, 8
cor, 7, 8, 10, 12

get_sample, 3, 5, 9
group_variables, 10

hclust, 7, 8, 10, 12
hierarchical_importance, 10

lime (aspect_importance), 2
list_variables, 12

model_triplet (calculate_triplet), 6

plot.aspect_importance, 13
plot.cluster_variables, 14
plot.hierarchical_importance
 (hierarchical_importance), 10
plot.triplet, 15
predict_aspects (aspect_importance), 2
predict_triplet (calculate_triplet), 6
print.aspect_importance, 16
print.triplet (calculate_triplet), 6