Package: corrgrapher (via r-universe)

September 18, 2024

Type Package

Title Explore Correlations Between Variables in a Machine Learning
Model

Version 1.0.2

Encoding UTF-8

Maintainer Pawel Morgen <seriousmorgen@protonmail.com>

Description When exploring data or models we often examine variables
one by one. This analysis is incomplete if the relationship
between these variables is not taken into account. The
'corrgrapher' package facilitates simultaneous exploration of
the Partial Dependence Profiles and the correlation between
variables in the model. The package 'corrgrapher' is a part of
the 'DrWhy.AI' universe.

License GPL (>= 2)

Depends R (>= 3.5.0)

Imports visNetwork, DALEX, ingredients, htmltools, jsonlite, ggplot2,
knitr

Suggests rmarkdown, testthat, gbm, ranger, spelling, covr

RoxygenNote 7.1.1

URL https://modeloriented.github.io/corrgrapher/,
https://github.com/ModelOriented/corrgrapher

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

Language en-US

Repository https://modeloriented.r-universe.dev

RemoteUrl https://github.com/modeloriented/corrgrapher

RemoteRef HEAD

RemoteSha 65ff19e10899e2f057e4634054e7908150c0e02c

Contents

Index **[10](#)**

---

calculate_cors	*Calculate correlation coefficients*

Description

Calculate correlation coefficients between variables in a data.frame, matrix or table using 3 different functions for 3 different possible pairs of vairables:

- numeric - numeric

- numeric - categorical

- categorical - categorical

Usage

```
calculate_cors(
  x,
  num_num_f = NULL,
  num_cat_f = NULL,
  cat_cat_f = NULL,
  max_cor = NULL
)

## S3 method for class 'explainer'
calculate_cors(
  x,
  num_num_f = NULL,
  num_cat_f = NULL,
  cat_cat_f = NULL,
  max_cor = NULL
)

## S3 method for class 'matrix'
calculate_cors(
  x,
  num_num_f = NULL,
  num_cat_f = NULL,
  cat_cat_f = NULL,
```

```
  max_cor = NULL
)

## S3 method for class 'table'
calculate_cors(
  x,
  num_num_f = NULL,
  num_cat_f = NULL,
  cat_cat_f = NULL,
  max_cor = NULL
)

## Default S3 method:
calculate_cors(
  x,
  num_num_f = NULL,
  num_cat_f = NULL,
  cat_cat_f = NULL,
  max_cor = NULL
)
```

Arguments

x	object used to select method. See more below.
num_num_f	A `function` used to determine correlation coefficient between a pair of numeric variables
num_cat_f	A `function` used to determine correlation coefficient between a pair of numeric and categorical variable
cat_cat_f	A `function` used to determine correlation coefficient between a pair of categorical variables
max_cor	A number used to indicate absolute correlation (like 1 in `cor`). Must be supplied if any of `*_f` arguments is supplied.

Value

A symmetrical `matrix` A of size n x n, where n - amount of columns in x (or dimensions for `table`). The value at A(i,j) is the correlation coefficient between ith and jth variable. On the diagonal, values from `max_cor` are set.

X argument

When x is a `data.frame`, all columns of `numeric` type are treated as numeric variables and all columns of `factor` type are treated as categorical variables. Columns of other types are ignored.

When x is a `matrix`, it is converted to `data.frame` using [as.data.frame.matrix](#).

When x is a `explainer`, the tests are performed on its `data` element.

When x is a `table`, it is treated as contingency table. Its dimensions must be named, but none of them may be named `Frequency`.

Default functions

By default, the function calculates p_value of statistical tests (cor.test for 2 numeric, chisq.test for factor and kruskal.test for mixed).

Then, the correlation coefficients are calculated as -log10(p_value). Any results above 100 are treated as absolute correlation and cut to 100.

The results are then divided by 100 to fit inside [0,1].

If only numeric data was supplied, the function used is cor.test.

Custom functions

Creating consistent measures for correlation coefficients, which are comparable for different kinds of variables, is a non-trivial task. Therefore, if user wishes to use custom function for calculating correlation coefficients, he must provide **all** necessary functions. Using a custom function for one case and a default for the other is consciously not supported. Naturally, user may supply copies of default functions at his own responsibility.

Function calculate_cors chooses, which parameters of *_f are required based on data supported. For example, for a matrix with numeric data only num_num_f is required. On the other hand, for a table only cat_cat_f is required.

All *_f parameters must be functions, which accept 2 parameters (numeric or factor vectors respectively) and return a single number from [0,max_num]. The num_cat_f must accept numeric argument as first and factor argument as second.

See Also

cor.test, chisq.test, kruskal.test

Examples

```
data(mtcars)
# Make sure, that categorical variables are factors
mtcars$vs <- factor(mtcars$vs, labels = c('V-shaped', 'straight'))
mtcars$am <- factor(mtcars$am, labels = c('automatic', 'manual'))
calculate_cors(mtcars)

# For a table:
data(HairEyeColor)
calculate_cors(HairEyeColor)

# Custom functions:
num_mtcars <- mtcars[,-which(colnames(mtcars) %in% c('vs', 'am'))]
my_f <- function(x,y) cor.test(x, y, method = 'spearman', exact=FALSE)$estimate
calculate_cors(num_mtcars, num_num_f = my_f, max_cor = 1)
```

---

corrgrapher   *Create a* corrgrapher *object*

Description

This is the main function of corrgrapher package. It does necessary calculations and creates a corrgrapher object. Feel free to pass it into plot, include it in knitr report or generate a simple HTML.

Usage

```
corrgrapher(x, ...)

## S3 method for class 'explainer'
corrgrapher(
  x,
  cutoff = 0.2,
  values = NULL,
  cor_functions = list(),
  ...,
  feature_importance = NULL,
  partial_dependence = NULL
)

## S3 method for class 'matrix'
corrgrapher(x, cutoff = 0.2, values = NULL, cor_functions = list(), ...)

## Default S3 method:
corrgrapher(x, cutoff = 0.2, values = NULL, cor_functions = list(), ...)
```

Arguments

x	an object to be used to select the method, which must satisfy conditions:
	• if data.frame (default), columns of numeric type must contain numerical variables and columns of factor class must contain categorical variables. Columns of other types will be ignored.
	• if explainer, methods [feature_importance](#) and [partial_dependence](#) must not return an error. See also arguments feature_importance and partial_dependence.
	• if matrix, it will be converted with [as.data.frame](#).
...	other arguments.
cutoff	a number. Correlations below this are treated as **no** correlation. Edges corresponding to them will **not** be included in the graph.
values	a data.frame with information about size of the nodes, containing columns value and label (consistent with colnames of x). Default set to equal for all nodes, or (for explainer) importance of variables.

cor_functions   a named list of functions to pass to calculate_cors. Must contain neces-
                   sary functions from num_num_f, num_cat_f or cat_cat_f. Must contain also
                   max_cor

feature_importance

                   Either:

                       • an object of feature importance_explainer class, created by feature_importance
                         function, or
                       • a named list of parameters to pass to feature_importance function.

partial_dependence

                   a named list with 2 elements: numerical and categorical. Both of them
                   should be either:

                       • an object of aggregated_profile_explainer class, created by partial_dependence
                         function, or
                       • a named list of parameters to pass to partial_dependence.

                   If only one kind of data was used, use a list with 1 object.

Details

Data analysis (and creating ML models) involves many stages. For early exploration, it is useful to
have a grip not only on individual series (AKA variables) available, but also on relations between
them. Unfortunately, the task of understanding correlations between variables proves to be difficult.
corrgrapher package aims to plot correlations between variables in form of a graph. Each node on
it is associated with single variable. Variables correlated with each other (positively and negatively
alike) shall be close, and weakly correlated - far from each other.

Value

A corrgrapher object. Essentially a list, consisting of following fields:

   • nodes - a data.frame to pass as argument nodes to visNetwork function

   • edges - a data.frame to pass as argument edges to visNetwork function

   • pds (if x was of explainer class) - a list with 2 elements: numerical and categorical.
     Each of them contains an object of aggregated_profiles_explainer used to create partial
     dependency plots.

   • data - data used to create the object.

See Also

plot.corrgrapher, knit_print.corrgrapher, save_to_html

Examples

```
# convert the category variable
df <- as.data.frame(datasets::Seatbelts)
df$law <- factor(df$law)
cgr <- corrgrapher(df)
```

knit_print.corrgrapher

Knitr S3 method

Description

This method allows corrgrapher objects to be displayed nicely in knitr/rmarkdown documents.

Usage

```
## S3 method for class 'corrgrapher'
knit_print(x, ...)
```

Arguments

x   An object of corrgrapher class. See corrgrapher function.

...   Other parameters, passed directly to knit_print.shiny.tag

Value

2 objects will be displayed: graph of correlations on the left and a plot on the right. If x was created from explainer, the plot will visualize partial dependency of the currently selected variable. In other case, the plot will visualize distribution of the variable.

plot.corrgrapher   *Visualize correlations in a corrgrapher object*

Description

Visualize correlations between variables, using previously created corrgrapher object.

Usage

```
## S3 method for class 'corrgrapher'
plot(x, ...)
```

Arguments

x   a corrgrapher object. See corrgrapher.

...   other parameters, passed directly to visNetwork function (such as main, submain, width, height etc.)

Value

A visNetwork object; graph. On this graph, the edges are treated as springs. The variables correlated **strongly** (positively or negatively) are **close** to each other, and those not (or weakly) correlated - **far** from each other.

See Also

corrgrapher

Examples

```
df <- as.data.frame(datasets::Seatbelts)[,1:7] # drop the binary target variable
cgr <- corrgrapher(df)
plot(cgr)
```

---

print.corrgrapher	*Print S3 method*

Description

This method allows corrgrapher objects to be displayed nicely in RStudio viewer.

Usage

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

Arguments

x	An object of corrgrapher class. See corrgrapher function.
...	Other parameters, passed directly to save_to_html function. If x was created from explainer, the plot will visualize partial dependency of the currently selected variable. In other case, the plot will visualize distribution of the variable.

---

save_to_html	*Generate and save HTML report*

Description

Create an interactive document in HTML based on corrgrapher object.

Usage

```
save_to_html(cgr, file = "report.html", overwrite = FALSE, ...)
```

Arguments

cgr	An object of corrgrapher class. See corrgrapher function.
file	File to write content to; passed directly to save_html.
overwrite	If file exists, should it be overwritten?
...	Other parameters

Value

A file of `file` name will be generated with 2 elements: graph of correlations in the middle and a plot on the right. If x was created from `explainer`, the plot will visualize partial dependency of the currently selected variable. In other case, the plot will visualize distribution of the variable.

Index