Package: rSAFE (via r-universe)

September 9, 2024

Title Surrogate-Assisted Feature Extraction

Version 0.1.4

Description Provides a model agnostic tool for white-box model trained
on features extracted from a black-box model. For more
information see: Gosiewska et al. (2020)
<doi:10.1016/j.dss.2021.113556>.

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.1

Imports DALEX, dendextend, ggplot2, ggpubr, grDevices, ingredients,
sets, stats

Suggests gbm, knitr, pander, randomForest, rmarkdown, spelling,
testthat, vdiffr

VignetteBuilder knitr

URL `https://github.com/ModelOriented/rSAFE`

BugReports `https://github.com/ModelOriented/rSAFE/issues`

Language en-US

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

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

RemoteRef HEAD

RemoteSha 4e9ad1a9dbac14b50d0dd3cd711adb294e0bf456

Contents

`apartments` *Apartments data*

Description

Datasets `apartments` and `apartmentsTest` are artificial, generated from the same model. Structure of the dataset is copied from real dataset from `PBImisc` package, but they were generated in a way to mimic effect of Anscombe quartet for complex black box models.

Usage

```
data(apartments)
```

Format

a data frame with 1000 rows and 6 columns

Details

- m2.price - price per square meter
- surface - apartment area in square meters
- no.rooms - number of rooms (correlated with surface)
- district - district in which apartment is located, factor with 10 levels (Bemowo, Bielany, Mokotow, Ochota, Praga, Srodmiescie, Ursus, Ursynow, Wola, Zoliborz)
- floor - floor
- construction.year - construction year

`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.safe_extractor` *Plotting Transformations of the SAFE Extractor Object*

Description

Plotting Transformations of the SAFE Extractor Object

Usage

```
## S3 method for class 'safe_extractor'
plot(x, ..., variable = NULL)
```

Arguments

`x`	safe_extractor object containing information about variables transformations created with safe_extraction() function
`...`	other parameters
`variable`	character, name of the variable to be plotted

Value

a plot object

`print.safe_extractor` *Printing Summary of the SAFE Extractor Object*

Description

Printing Summary of the SAFE Extractor Object

Usage

```
## S3 method for class 'safe_extractor'
print(x, ..., variable = NULL)
```

Arguments

`x`	safe_extractor object containing information about variables transformations created with safe_extraction() function
`...`	other parameters
`variable`	character, name of the variable to be plotted. If this argument is not specified then transformations for all variables are printed

Value

No return value, prints the structure of the object

---

safely_detect_changepoints *Identifying Changes in a Series Using PELT Algorithm*

Description

The safely_detect_changepoints() function calculates the optimal positioning and number of changepoints for given data and penalty. It uses a PELT algorithm with a nonparametric cost function based on the empirical distribution. The implementation is inspired by the code available on https://github.com/rkillick/changepoint.

Usage

```
safely_detect_changepoints(data, penalty = "MBIC", nquantiles = 10)
```

Arguments

`data`	a vector within which you wish to find changepoints
`penalty`	penalty for introducing another changepoint, one of "AIC", "BIC", "SIC", "MBIC", "Hannan-Quinn" or numeric non-negative value
`nquantiles`	the number of quantiles used in integral approximation

Value

a vector of optimal changepoint positions (last observations of each segment)

See Also

`safely_transform_continuous`

Examples

```
library(rSAFE)

data <- rep(c(2,7), each=4)
safely_detect_changepoints(data)

set.seed(123)
data <- c(rnorm(15, 0), rnorm(20, 2), rnorm(30, 8))
safely_detect_changepoints(data)
safely_detect_changepoints(data, penalty = 25)
```

safely_detect_interactions

Detecting Interactions via Permutation Approach

Description

The safely_detect_interactions() function detects second-order interactions based on predictions made by a surrogate model. For each pair of features it performs values permutation in order to evaluate their non_additive effect.

Usage

```
safely_detect_interactions(
  explainer,
  inter_param = 0.5,
  inter_threshold = 0.5,
  verbose = TRUE
)
```

Arguments

`explainer`	DALEX explainer created with explain() function
`inter_param`	numeric, a positive value indicating which of single observation non-additive effects are to be regarded as significant, the higher value the higher non-additive effect has to be to be taken into account
`inter_threshold`	numeric, a value from `[0,1]` interval indicating which interactions should be returned as significant. It corresponds to the percentage of observations for which interaction measure is greater than inter_param - if this percentage is less than inter_threshold then interaction effect is ignored.
`verbose`	logical, if progress bar is to be printed

Value

dataframe object containing interactions effects greater than or equal to the specified inter_threshold

See Also

safe_extraction

Examples

```
library(DALEX)
library(randomForest)
library(rSAFE)

data <- apartments[1:500,]
```

```
set.seed(111)
model_rf <- randomForest(m2.price ~ construction.year + surface + floor +
                           no.rooms + district, data = data)
explainer_rf <- explain(model_rf, data = data[,2:6], y = data[,1])
safely_detect_interactions(explainer_rf, inter_param = 0.25,
                          inter_threshold = 0.2, verbose = TRUE)
```

---

safely_select_variables

Performing Feature Selection on the Dataset with Transformed Variables

---

Description

The safely_select_variables() function selects variables from dataset returned by safely_transform_data() function. For each original variable exactly one variable is chosen

- either original one or transformed one. The choice is based on the AIC value for linear model (regression) or logistic regression (classification).

Usage

```
safely_select_variables(
  safe_extractor,
  data,
  y = NULL,
  which_y = NULL,
  class_pred = NULL,
  verbose = TRUE
)
```

Arguments

`safe_extractor`	object containing information about variables transformations created with safe_extraction() function
`data`	data, original dataset or the one returned by safely_transform_data() function. If data do not contain transformed variables then transformation is done inside this function using 'safe_extractor' argument. Data may contain response variable or not - if it does then 'which_y' argument must be given, otherwise 'y' argument should be provided.
`y`	vector of responses, must be given if data does not contain it
`which_y`	numeric or character (optional), must be given if data contains response values
`class_pred`	numeric or character, used only in multi-classification problems. If response vector has more than two levels, then 'class_pred' should indicate the class of interest which will denote failure - all other classes will stand for success.
`verbose`	logical, if progress bar is to be printed

Value

vector of variables names, selected based on AIC values

See Also

safely_transform_data

Examples

```
library(DALEX)
library(randomForest)
library(rSAFE)

data <- apartments[1:500,]
set.seed(111)
model_rf <- randomForest(m2.price ~ construction.year + surface + floor +
                           no.rooms + district, data = data)
explainer_rf <- explain(model_rf, data = data[,2:6], y = data[,1])
safe_extractor <- safe_extraction(explainer_rf, verbose = FALSE)
safely_select_variables(safe_extractor, data, which_y = "m2.price", verbose = FALSE)
```

---

safely_transform_categorical

Calculating a Transformation of Categorical Feature Using Hierarchical Clustering

---

Description

The safely_transform_categorical() function calculates a transformation function for the categorical variable using predictions obtained from black box model and hierarchical clustering. The gap statistic criterion is used to determine the optimal number of clusters.

Usage

```
safely_transform_categorical(
  explainer,
  variable,
  method = "complete",
  B = 500,
  collapse = "_"
)
```

Arguments

`explainer`	DALEX explainer created with explain() function
`variable`	a feature for which the transformation function is to be computed

method	the agglomeration method to be used in hierarchical clustering, one of: "ward.D", "ward.D2", "single", "complete", "average", "mcquitty", "median", "centroid"
B	number of reference datasets used to calculate gap statistics
collapse	a character string to separate original levels while combining them to the new one

Value

list of information on the transformation of given variable

See Also

safe_extraction

Examples

```
library(DALEX)
library(randomForest)
library(rSAFE)

data <- apartments[1:500,]
set.seed(111)
model_rf <- randomForest(m2.price ~ construction.year + surface + floor +
                         no.rooms + district, data = data)
explainer_rf <- explain(model_rf, data = data[,2:6], y = data[,1])
safely_transform_categorical(explainer_rf, "district")
```

---

safely_transform_continuous

Calculating a Transformation of a Continuous Feature Using PDP/ALE Plot

---

Description

The safely_transform_continuous() function calculates a transformation function for the continuous variable using a PD/ALE plot obtained from black box model.

Usage

```
safely_transform_continuous(
  explainer,
  variable,
  response_type = "ale",
  grid_points = 50,
  N = 200,
  penalty = "MBIC",
  nquantiles = 10,
  no_segments = 2
)
```

Arguments

explainer	DALEX explainer created with explain() function
variable	a feature for which the transformation function is to be computed
response_type	character, type of response to be calculated, one of: "pdp", "ale". If features are uncorrelated, one can use "pdp" type - otherwise "ale" is strongly recommended.
grid_points	number of points on x-axis used for creating the PD/ALE plot, default 50
N	number of observations from the dataset used for creating the PD/ALE plot, default 200
penalty	penalty for introducing another changepoint, one of "AIC", "BIC", "SIC", "MBIC", "Hannan-Quinn" or numeric non-negative value
nquantiles	the number of quantiles used in integral approximation
no_segments	numeric, a number of segments variable is to be divided into in case of founding no breakpoints

Value

list of information on the transformation of given variable

See Also

safe_extraction, safely_detect_changepoints

Examples

```
library(DALEX)
library(randomForest)
library(rSAFE)

data <- apartments[1:500,]
set.seed(111)
model_rf <- randomForest(m2.price ~ construction.year + surface + floor +
                           no.rooms + district, data = data)
explainer_rf <- explain(model_rf, data = data[,2:6], y = data[,1])
safely_transform_continuous(explainer_rf, "construction.year")
```

safely_transform_data *Performing Transformations on All Features in the Dataset*

Description

The safely_transform_data() function creates new variables in dataset using safe_extractor object.

Usage

```
safely_transform_data(safe_extractor, data, verbose = TRUE)
```

Arguments

`safe_extractor`	object containing information about variables transformations created with safe_extraction() function
`data`	data for which features are to be transformed
`verbose`	logical, if progress bar is to be printed

Value

data with extra columns containing newly created variables

See Also

`safe_extraction`, `safely_select_variables`

Examples

```
library(DALEX)
library(randomForest)
library(rSAFE)

data <- apartments[1:500,]
set.seed(111)
model_rf <- randomForest(m2.price ~ construction.year + surface + floor +
                           no.rooms + district, data = data)
explainer_rf <- explain(model_rf, data = data[,2:6], y = data[,1])
safe_extractor <- safe_extraction(explainer_rf, verbose = FALSE)
safely_transform_data(safe_extractor, data, verbose = FALSE)
```

`safe_extraction` *Creating SAFE Extractor - an Object Used for Surrogate-Assisted Feature Extraction*

Description

The safe_extraction() function creates a SAFE-extractor object which may be used later for surrogate feature extraction.

Usage

```
safe_extraction(
  explainer,
  response_type = "ale",
  grid_points = 50,
  N = 200,
  penalty = "MBIC",
  nquantiles = 10,
  no_segments = 2,
  method = "complete",
  B = 500,
  collapse = "_",
  interactions = FALSE,
  inter_param = 0.25,
  inter_threshold = 0.25,
  verbose = TRUE
)
```

Arguments

`explainer`	DALEX explainer created with explain() function
`response_type`	character, type of response to be calculated, one of: "pdp", "ale". If features are uncorrelated, one can use "pdp" type - otherwise "ale" is strongly recommended.
`grid_points`	number of points on x-axis used for creating the PD/ALE plot, default 50
`N`	number of observations from the dataset used for creating the PD/ALE plot, default 200
`penalty`	penalty for introducing another changepoint, one of "AIC", "BIC", "SIC", "MBIC", "Hannan-Quinn" or numeric non-negative value
`nquantiles`	the number of quantiles used in integral approximation
`no_segments`	numeric, a number of segments variable is to be divided into in case of founding no breakpoints
`method`	the agglomeration method to be used in hierarchical clustering, one of: "ward.D", "ward.D2", "single", "complete", "average", "mcquitty", "median", "centroid"
`B`	number of reference datasets used to calculate gap statistics
`collapse`	a character string to separate original levels while combining them to the new one
`interactions`	logical, if interactions between variables are to be taken into account
`inter_param`	numeric, a positive value indicating which of single observation non-additive effects are to be regarded as significant, the higher value the higher non-additive effect has to be to be taken into account
`inter_threshold`	numeric, a value from `[0,1]` interval indicating which interactions should be returned as significant. It corresponds to the percentage of observations for which interaction measure is greater than inter_param - if this percentage is less than inter_threshold then interaction effect is ignored.
`verbose`	logical, if progress bar is to be printed

Value

safe_extractor object containing information about variables transformation

See Also

`safely_transform_categorical`, `safely_transform_continuous`, `safely_detect_interactions`, `safely_transform_data`

Examples

```
library(DALEX)
library(randomForest)
library(rSAFE)

data <- apartments[1:500,]
set.seed(111)
model_rf <- randomForest(m2.price ~ construction.year + surface + floor +
                           no.rooms + district, data = data)
explainer_rf <- explain(model_rf, data = data[,2:6], y = data[,1], verbose = FALSE)
safe_extractor <- safe_extraction(explainer_rf, grid_points = 30, N = 100, verbose = FALSE)
print(safe_extractor)
plot(safe_extractor, variable = "construction.year")
```

Index