Package 'EIX'

July 21, 2025

Title Explain Interactions in 'XGBoost'

Version 1.2.0

Description Structure mining from 'XGBoost' and 'LightGBM' models.
Key functionalities of this package cover: visualisation of tree-based ensembles models,
identification of interactions, measuring of variable importance,
measuring of interaction importance, explanation of single prediction
with break down plots (based on 'xgboostExplainer' and 'iBreakDown' packages).
To download the 'LightGBM' use the following link: <https://github.com/Microsoft/LightGBM>.
'EIX' is a part of the 'DrWhy.AI' universe.

Depends R (>= 3.5.0)

License GPL-2

Encoding UTF-8

LazyData true

Imports MASS, ggplot2, data.table, purrr, xgboost, DALEX, ggrepel,
ggiraphExtra, iBreakDown, tidyr, scales

RoxygenNote 7.1.1

Suggests Matrix, knitr, rmarkdown, lightgbm

VignetteBuilder knitr

URL https://github.com/ModelOriented/EIX

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

NeedsCompilation no

Author Szymon Maksymiuk [aut, cre],
Ewelina Karbowiak [aut],
Przemyslaw Biecek [aut, ths]

Maintainer Szymon Maksymiuk <sz.maksymiuk@gmail.com>

Repository CRAN

Date/Publication 2021-03-23 08:10:02 UTC

Contents

EIX-package . . . 2
HR_data . . . 2
importance . . . 3
interactions . . . 5
lollipop . . . 6
plot.importance . . . 7
plot.interactions . . . 9
plot.lollipop . . . 10
titanic_data . . . 12
waterfall . . . 13

Index **15**

`EIX-package` *EIX package*

Description

Structure mining from 'XGBoost' and 'LightGBM' models. Key functionalities of this package cover: visualisation of tree-based ensembles models, identification of interactions, measuring of variable importance, measuring of interaction importance, explanation of single prediction with break down plots (based on 'xgboostExplainer' and 'iBreakDown' packages). To download the 'LightGBM' use the following link: <https://github.com/Microsoft/LightGBM>. EIX' is a part of the 'DrWhy.AI' universe.

`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/ludobenistant/hr-analytics/data

Format

A data table with 14999 rows and 10 variables

Details

The description of the dataset was copied from the `breakDown` package.

- 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_montly_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

https://www.kaggle.com/ludobenistant/hr-analytics/data, https://cran.r-project.org/package=breakDown

---

`importance` *Importance of variables and interactions in the model*

Description

This functions calculates a table with selected measures of importance for variables and interactions.

Usage

```
importance(xgb_model, data, option = "both", digits = 4)
```

Arguments

`xgb_model`	a xgboost or lightgbm model.
`data`	a data table with data used to train the model.
`option`	if "variables" then table includes only single variables, if "interactions", then only interactions if "both", then both single variable and interactions. Default "both".
`digits`	number of significant digits that shall be returned. Will be passed to the signif() functions.

Details

Available measures:

- "sumGain" - sum of Gain value in all nodes, in which given variable occurs,
- "sumCover" - sum of Cover value in all nodes, in which given variable occurs; for LightGBM models: number of observation, which pass through the node,
- "mean5Gain" - mean gain from 5 occurrences of given variable with the highest gain,
- "meanGain" - mean Gain value in all nodes, in which given variable occurs,
- "meanCover" - mean Cover value in all nodes, in which given variable occurs; for LightGBM models: mean number of observation, which pass through the node,
- "freqency" - number of occurrences in the nodes for given variable.

Additionally for table with single variables:

- "meanDepth" - mean depth weighted by gain,
- "numberOfRoots" - number of occurrences in the root,
- "weightedRoot" - mean number of occurrences in the root, which is weighted by gain.

Value

a data table

Examples

```
library("EIX")
library("Matrix")
sm <- sparse.model.matrix(left ~ . - 1,  data = HR_data)

library("xgboost")
param <- list(objective = "binary:logistic", max_depth = 2)
xgb_model <- xgboost(sm, params = param, label = HR_data[, left] == 1, nrounds = 25, verbose=0)

imp <- importance(xgb_model, sm, option = "both")
imp
plot(imp,  top = 10)

imp <- importance(xgb_model, sm, option = "variables")
imp
plot(imp,  top = nrow(imp))

 imp <- importance(xgb_model, sm, option = "interactions")
 imp
plot(imp,  top =  nrow(imp))

 imp <- importance(xgb_model, sm, option = "variables")
 imp
plot(imp, top = NULL, radar = FALSE, xmeasure = "sumCover", ymeasure = "sumGain")
```

`interactions` *Importance of interactions and pairs in the model*

Description

This function calculates a table with two measures of importance for interactions and pairs in the model.

Usage

```
interactions(xgb_model, data, option = "interactions")
```

Arguments

`xgb_model`	a xgboost or lightgbm model.
`data`	a data table with data used to train the model.
`option`	if "interactions", the table contains interactions, if "pairs", this table contains all the pairs in the model. Default "interactions".

Details

Available measures:

- "sumGain" - sum of Gain value in all nodes, in which given variable occurs,
- "freqency" - number of occurrences in the nodes for given variable.

NOTE: Be careful use of this function with `option="pairs"` parameter, because high gain of pair can be a result of high gain of child variable. As strong interactions should be considered only these pairs of variables, where variable on the bottom (child) has higher gain than variable on the top (parent).

Value

a data table

Examples

```
library("EIX")
library("Matrix")
sm <- sparse.model.matrix(left ~ . - 1,  data = HR_data)

library("xgboost")
param <- list(objective = "binary:logistic", max_depth = 2)
xgb_model <- xgboost(sm, params = param, label = HR_data[, left] == 1, nrounds = 25, verbose=0)

inter <- interactions(xgb_model, sm, option = "interactions")
inter
plot(inter)
```

```
inter <- interactions(xgb_model, sm, option = "pairs")
inter
plot(inter)


library(lightgbm)
train_data <- lgb.Dataset(sm, label =  HR_data[, left] == 1)
params <- list(objective = "binary", max_depth = 2)
lgb_model <- lgb.train(params, train_data, 25)

inter <- interactions(lgb_model, sm, option = "interactions")
inter
plot(inter)

inter <- interactions(lgb_model, sm, option = "pairs")
inter
plot(inter)
```

`lollipop` *Tables needed for lollipop plot*

Description

This function calculates two tables needed to generate lollipop plot, which visualise the model. The first table contains information about all nodes in the trees forming a model. It includes gain value, depth and ID of each nodes. The second table contains similarly information about roots in the trees.

Usage

```
lollipop(xgb_model, data)
```

Arguments

`xgb_model`	a xgboost or lightgbm model.
`data`	a data table with data used to train the model.

Value

an object of the lollipop class

Examples

```
library("EIX")
library("Matrix")
sm <- sparse.model.matrix(left ~ . - 1,  data = HR_data)

library("xgboost")
param <- list(objective = "binary:logistic", max_depth = 2)
xgb_model <- xgboost(sm, params = param, label = HR_data[, left] == 1, nrounds = 25, verbose = 0)

lolli <- lollipop(xgb_model, sm)
plot(lolli, labels = "topAll", log_scale = TRUE)


library(lightgbm)
train_data <- lgb.Dataset(sm, label =  HR_data[, left] == 1)
params <- list(objective = "binary", max_depth = 2)
lgb_model <- lgb.train(params, train_data, 25)

lolli <- lollipop(lgb_model, sm)
plot(lolli, labels = "topAll", log_scale = TRUE)
```

---

plot.importance *Plot importance measures*

Description

This functions plots selected measures of importance for variables and interactions. It is possible to visualise importance table in two ways: radar plot with six measures and scatter plot with two choosen measures.

Usage

```
## S3 method for class 'importance'
plot(
  x,
  ...,
  top = 10,
  radar = TRUE,
  text_start_point = 0.5,
  text_size = 3.5,
  xmeasure = "sumCover",
  ymeasure = "sumGain"
)
```

Arguments

x	a result from the importance function.
...	other parameters.
top	number of positions on the plot or NULL for all variable. Default 10.
radar	TRUE/FALSE. If TRUE the plot shows six measures of variables' or interactions' importance in the model. If FALSE the plot containing two chosen measures of variables' or interactions' importance in the model.
text_start_point	place, where the names of the particular feature start. Available for 'radar=TRUE'. Range from 0 to 1. Default 0.5.
text_size	size of the text on the plot. Default 3.5.
xmeasure	measure on the x-axis.Available for 'radar=FALSE'. Default "sumCover".
ymeasure	measure on the y-axis. Available for 'radar=FALSE'. Default "sumGain".

Details

Available measures:

- "sumGain" - sum of Gain value in all nodes, in which given variable occurs,
- "sumCover" - sum of Cover value in all nodes, in which given variable occurs; for LightGBM models: number of observation, which pass through the node,
- "mean5Gain" - mean gain from 5 occurrences of given variable with the highest gain,
- "meanGain" - mean Gain value in all nodes, in which given variable occurs,
- "meanCover" - mean Cover value in all nodes, in which given variable occurs; for LightGBM models: mean number of observation, which pass through the node,
- "freqency" - number of occurrences in the nodes for given variable.

Additionally for plots with single variables:

- "meanDepth" - mean depth weighted by gain,
- "numberOfRoots" - number of occurrences in the root,
- "weightedRoot" - mean number of occurrences in the root, which is weighted by gain.

Value

a ggplot object

Examples

```
library("EIX")
library("Matrix")
sm <- sparse.model.matrix(left ~ . - 1,  data = HR_data)

library("xgboost")
param <- list(objective = "binary:logistic", max_depth = 2)
xgb_model <- xgboost(sm, params = param, label = HR_data[, left] == 1, nrounds = 25, verbose=0)
```

```
imp <- importance(xgb_model, sm, option = "both")
imp
plot(imp,  top = 10)

imp <- importance(xgb_model, sm, option = "variables")
imp
plot(imp,  top = nrow(imp))

 imp <- importance(xgb_model, sm, option = "interactions")
 imp
plot(imp,  top =  nrow(imp))

 imp <- importance(xgb_model, sm, option = "variables")
 imp
plot(imp, top = NULL, radar = FALSE, xmeasure = "sumCover", ymeasure = "sumGain")


library(lightgbm)
train_data <- lgb.Dataset(sm, label =  HR_data[, left] == 1)
params <- list(objective = "binary", max_depth = 2)
lgb_model <- lgb.train(params, train_data, 25)

imp <- importance(lgb_model, sm, option = "both")
imp
plot(imp,  top = nrow(imp))

imp <- importance(lgb_model, sm, option = "variables")
imp
plot(imp, top = NULL, radar = FALSE, xmeasure = "sumCover", ymeasure = "sumGain")
```

`plot.interactions` *Plot importance of interactions or pairs*

Description

This function plots the importance ranking of interactions and pairs in the model.

Usage

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

Arguments

`x`	a result from the `interactions` function.
`...`	other parameters.

Details

NOTE: Be careful use of this function with `option="pairs"` parameter, because high gain of pair can be a result of high gain of child variable. As strong interactions should be considered only these pairs of variables, where variable on the bottom (child) has higher gain than variable on the top (parent).

Value

a ggplot object

Examples

```
library("EIX")
library("Matrix")
sm <- sparse.model.matrix(left ~ . - 1,  data = HR_data)

library("xgboost")
param <- list(objective = "binary:logistic", max_depth = 2)
xgb_model <- xgboost(sm, params = param, label = HR_data[, left] == 1, nrounds = 25, verbose=0)

inter <- interactions(xgb_model, sm,option = "interactions")
inter
plot(inter)

inter <- interactions(xgb_model, sm,option = "pairs")
inter
plot(inter)


library(lightgbm)
train_data <- lgb.Dataset(sm, label =  HR_data[, left] == 1)
params <- list(objective = "binary", max_depth = 2)
lgb_model <- lgb.train(params, train_data, 25)

inter <- interactions(lgb_model, sm,option = "interactions")
inter
plot(inter)

inter <- interactions(lgb_model, sm,option = "pairs")
inter
plot(inter)
```

---

plot.lollipop *Visualiation of the model*

Description

The lollipop plots the model with the most important interactions and variables in the roots.

Usage

```
## S3 method for class 'lollipop'
plot(x, ..., labels = "topAll", log_scale = TRUE, threshold = 0.1)
```

Arguments

x	a result from the lollipop function.
...	other parameters.
labels	if "topAll" then labels for the most important interactions (vertical label) and variables in the roots (horizontal label) will be displayed, if "interactions" then labels for all interactions, if "roots" then labels for all variables in the root.
log_scale	TRUE/FALSE logarithmic scale on the plot. Default TRUE.
threshold	on the plot will occur only labels with Gain higher than 'threshold' of the max Gain value in the model. The lower threshold, the more labels on the plot. Range from 0 to 1. Default 0.1.

Value

a ggplot object

Examples

```
library("EIX")
library("Matrix")
sm <- sparse.model.matrix(left ~ . - 1,  data = HR_data)

library("xgboost")
param <- list(objective = "binary:logistic", max_depth = 2)
xgb_model <- xgboost(sm, params = param, label = HR_data[, left] == 1, nrounds = 25, verbose = 0)

lolli <- lollipop(xgb_model, sm)
plot(lolli, labels = "topAll", log_scale = TRUE)


library(lightgbm)
train_data <- lgb.Dataset(sm, label =  HR_data[, left] == 1)
params <- list(objective = "binary", max_depth = 3)
lgb_model <- lgb.train(params, train_data, 25)

lolli <- lollipop(lgb_model, sm)
plot(lolli, labels = "topAll", log_scale = TRUE)
```

`titanic_data` *Passengers and Crew on the RMS Titanic*

Description

The `titanic` data is a complete list of passengers and crew members on the RMS Titanic. It includes a variable indicating whether a person did survive the sinking of the RMS Titanic on April 15, 1912.

Usage

```
data(titanic_data)
```

Format

a data frame with 2207 rows and 11 columns

Details

The description of the dataset was copied from the `DALEX` package.

This dataset was copied from the `stablelearner` package and went through few variable transformations. Levels in `embarked` was replaced with full names, `sibsp`, `parch` and `fare` were converted to numerical variables and values for crew were replaced with 0. If you use this dataset please cite the original package.

From `stablelearner`: The website https://www.encyclopedia-titanica.org offers detailed information about passengers and crew members on the RMS Titanic. According to the website 1317 passengers and 890 crew member were abord. 8 musicians and 9 employees of the shipyard company are listed as passengers, but travelled with a free ticket, which is why they have `NA` values in `fare`. In addition to that, `fare` is truely missing for a few regular passengers.

- gender a factor with levels `male` and `female`.
- age a numeric value with the persons age on the day of the sinking.
- class a factor specifying the class for passengers or the type of service aboard for crew members.
- embarked a factor with the persons place of of embarkment (Belfast/Cherbourg/Queenstown/Southampton).
- country a factor with the persons home country.
- fare a numeric value with the ticket price (`0` for crew members, musicians and employees of the shipyard company).
- sibsp an ordered factor specifying the number if siblings/spouses aboard; adopted from Vanderbild data set (see below).
- parch an ordered factor specifying the number of parents/children aboard; adopted from Vanderbild data set (see below).
- survived a factor with two levels (`no` and `yes`) specifying whether the person has survived the sinking.

Source

The description of dataset was copied from the DALEX package. This dataset was copied from the stablelearner package and went through few variable transformations. The complete list of persons on the RMS titanic was downloaded from https://www.encyclopedia-titanica.org on April 5, 2016.

References

https://www.encyclopedia-titanica.org, https://CRAN.R-project.org/package=stablelearner, https://cran.r-project.org/package=DALEX.

`waterfall` *Explain prediction of a single observation*

Description

This function calculates a table with influence of variables and interactions on the prediction of a given observation. It supports only xgboost models.

Usage

```
waterfall(
  xgb_model,
  new_observation,
  data,
  type = "binary",
  option = "interactions",
  baseline = 0
)
```

Arguments

xgb_model	a xgboost model.
new_observation	a new observation.
data	row from the original dataset with the new observation to explain (not one-hot-encoded). The param above has to be set to merge categorical features. If you dont wont to merge categorical features, set this parameter the same as new_observation.
type	the learning task of the model. Available tasks: "binary" for binary classification or "regression" for linear regression.
option	if "variables", the plot includes only single variables, if "interactions", then only interactions. Default "interaction".
baseline	a number or a character "Intercept" (for model intercept). The baseline for the plot, where the rectangles should start. Default 0.

Details

The function contains code or pieces of code from `breakDown` code created by Przemysław Biecek and `xgboostExplainer` code created by David Foster.

Value

an object of the broken class

Examples

```
library("EIX")
library("Matrix")
sm <- sparse.model.matrix(left ~ . - 1,  data = HR_data)

library("xgboost")
param <- list(objective = "binary:logistic", max_depth = 2)
xgb_model <- xgboost(sm, params = param, label = HR_data[, left] == 1, nrounds = 25, verbose=0)

data <- HR_data[9,-7]
new_observation <- sm[9,]

wf <- waterfall(xgb_model, new_observation, data,  option = "interactions")
wf

plot(wf)
```

Index

∗ **titanic_data**
titanic_data, 12

EIX-package, 2

HR_data, 2

importance, 3
interactions, 5

lollipop, 6

plot.importance, 7
plot.interactions, 9
plot.lollipop, 10

titanic_data, 12

waterfall, 13