Package 'xspliner'

September 25, 2019

Title Assisted Model Building, using Surrogate Black-Box Models to Train Interpretable Spline Based Additive Models

Version 0.0.4

Description Builds generalized linear model with automatic data transformation. The 'xspliner' helps to build simple, interpretable models that inherits informations provided by more complicated ones. The resulting model may be treated as explanation of provided black box, that was supplied prior to the algorithm.

Depends R (>= 3.0)

License GPL

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports stats, pdp, dplyr, ggplot2, mgcv, magrittr, purrr, tidyr, pROC (>= 1.15.3)

Suggests ALEPlot, factorMerger, testthat, knitr, rmarkdown, ResourceSelection, randomForest, e1071, caret, breakDown, DALEX, xgboost, gridExtra, grid, ISLR

VignetteBuilder knitr

URL https://ModelOriented.github.io/xspliner/

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

NeedsCompilation no

Author Krystian Igras [aut, cre], Przemyslaw Biecek [aut, ths]

Maintainer Krystian Igras <krystian8207@gmail.com>

Repository CRAN

Date/Publication 2019-09-25 20:20:02 UTC

R topics documented:

`xspliner-package` *Easy way for approximating data with splines.*

Description

Easy way for approximating data with splines.

`approx_with_spline` *Approximate spline on data*

Description

It approximates data with spline function by fitting GAM model.

Usage

```
approx_with_spline(effect_data, response, predictor,
  env = parent.frame(), ...)

approx_with_monotonic_spline(effect_data, response, predictor,
  env = parent.frame(), monotonic, ...)
```

Arguments

effect_data	Black box response data, for example pdp curve.
response	Name of response value from effect_data.
predictor	Name of predictor value from effect_data.
env	Formula environment that should be used for fitting gam model.
...	Other arguments passed to s function.
monotonic	Possible options "up", "down" and "auto. If up the spline is increasing, when down decreasing.

Value

Object of class "gam". See gamObject

Examples

```
x <- sort(rnorm(20, 5, 5))
y <- rnorm(20, 2, 2)
env <- new.env()
approx_with_spline(data.frame(x = x, y = y), "y", "x", env)

approx_with_monotonic_spline(data.frame(x = x, y = y), "y", "x", env, "up")
```

---

build_xspliner *Helper function for building GLM object with transformed variables.*

Description

Helper function for building GLM object with transformed variables.

Usage

```
build_xspliner(formula, model, data, xf_opts = xf_opts_default,
  xs_opts = xs_opts_default, link = "identity", family = "gaussian",
  env = parent.frame(), compare_stat = aic, control, ...)
```

Arguments

formula	xspliner-specific formula object. Check vignette("xspliner") for more details.
model	Predictive model. Basic model used for extracting predictors transformation.
data	Training data of model.
xf_opts	Formula parameters used for factor variable transformations inherited from factorMerger package.
xs_opts	Predictive model response method and approximation parameters used for quantitative.

link	Link function that should be used in final model. The passed is used when cannot be extracted from model. By default 'identity'. See family for possibilities.
family	Family of response variable that should be used in final model. The passed is used when cannot be extracted from model. By default 'gaussian'. See family for possibilities.
env	Environment in which optional variables passed into parameters are stored.
compare_stat	Function of linear model (lm function output). Statistic that measures if linear model is better that transformed one. See stats.
control	Fitting settings. See glm.control.
...	Another parameters passed from chosen method. Not used.

---

log_msg *Helper function to print out log messages*

Description

Helper function to print out log messages

Usage

```
log_msg(message)
```

Arguments

message	Message that should be printed to R console

---

plot.xspliner *Plot method for 'xspliner' model*

Description

The method provides all plotting methods offered by 'xspliner' package. See plot_variable_transition and plot_model_comparison for more details.

Usage

```
## S3 method for class 'xspliner'
plot(x, variable_names = NULL, model = NULL,
  plot_response = TRUE, plot_approx = TRUE, data = NULL,
  plot_data = FALSE, plot_deriv = FALSE, n_plots = 6,
  sort_by = NULL, use_coeff = TRUE, compare_with = list(),
  prediction_funs = list(function(object, newdata) predict(object,
  newdata)), ...)
```

Arguments

x	Object of class 'xspliner'.
variable_names	Names of predictors which transitions should be plotted.
model	Base model that xspliner is based on.
plot_response	If TRUE black box model response is drawn.
plot_approx	If TRUE black box model response approximation is drawn.
data	Training data used for building x model. Required for plot_data option and model comparing.
plot_data	If TRUE raw data is drawn.
plot_deriv	If TRUE derivative of approximation is showed on plot.
n_plots	Threshold for number of plots when plotting all variables.
sort_by	When comparing models determines according to which model should observations be ordered.
use_coeff	If TRUE both PDP function and its approximation is scaled with corresponding surrogate model coefficient.
compare_with	Named list. Other models that should be compared with xspliner and model.
prediction_funs	Prediction functions that should be used in model comparison.
...	Another arguments passed into model specific method.

---

plot_model_comparison *Plot models comparison*

Description

The function plots models comparison based on them predictions.

Usage

```
plot_model_comparison(x, model, data, compare_with = list(),
  prediction_funs = list(function(object, newdata) predict(object,
  newdata)), sort_by = NULL)
```

Arguments

x	Object of class 'xspliner'
model	Base model that xspliner is based on.
data	Dataset on which predictions should be compared.
compare_with	Named list. Other models that should be compared with xspliner and model.
prediction_funs	Prediction functions that should be used in model comparison.
sort_by	When comparing models determines according to which model should observations be ordered.

Examples

```
iris_data <- droplevels(iris[iris$Species != "setosa", ])
library(e1071)
library(randomForest)
library(xspliner)
# Build SVM model, random forest model and surrogate one constructed on top od SVM
model_svm <- svm(Species ~  Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
                 data = iris_data, probability = TRUE)
model_rf <- randomForest(
  Species ~  Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
  data = iris_data
)
model_xs <- xspline(
  Species ~  xs(Sepal.Length) + xs(Sepal.Width) + xs(Petal.Length) + xs(Petal.Width),
  model = model_svm
)
# Prepare prediction functions returning label probability
prob_svm <- function(object, newdata)
  attr(predict(object, newdata = newdata, probability = TRUE), "probabilities")[, 2]
prob_rf <- function(object, newdata)
  predict(object, newdata = newdata, type = "prob")[, 2]
prob_xs <- function(object, newdata)
  predict(object, newdata = newdata, type = "response")

# Plotting predictions for original SVM and surrogate model on training data
plot_model_comparison(
  model_xs, model_svm, data = iris_data,
  prediction_funs = list(xs = prob_xs, svm = prob_svm)
)
# Plotting predictions for original SVM, surrogate model and random forest on training data
plot_model_comparison(
  model_xs, model_svm, data = iris_data,
  compare_with = list(rf = model_rf),
  prediction_funs = list(xs = prob_xs, svm = prob_svm, rf = prob_rf)
)
# Sorting values according to SVM predictions
plot_model_comparison(
  model_xs, model_svm, data = iris_data,
  compare_with = list(rf = model_rf),
  prediction_funs = list(xs = prob_xs, svm = prob_svm, rf = prob_rf),
  sort_by = "svm"
)
```

`plot_variable_transition` *Plot variable profile*

Description

The function plots variable profile. In case of quantiitative variable it plots original transition function and its spline approximation. The function provides possibility to plot data points and transition derivative as well. In case of qualitative variable it plots merging path for variable levels. When no variable is specified it plots transitions for first `n_plots` variables.

Usage

```
plot_variable_transition(x, variable_names = NULL,
  plot_response = TRUE, plot_approx = TRUE, data = NULL,
  plot_data = FALSE, plot_deriv = FALSE, n_plots = 6,
  use_coeff = TRUE)
```

Arguments

`x`	Object of class 'xspliner'.
`variable_names`	Names of predictors which transitions should be plotted.
`plot_response`	If TRUE black box model response is drawn.
`plot_approx`	If TRUE black box model response approximation is drawn.
`data`	Training data used for building `x` model. Required for plot_data option.
`plot_data`	If TRUE raw data is drawn.
`plot_deriv`	If TRUE derivative of approximation is showed on plot.
`n_plots`	Threshold for number of plots when plotting all variables.
`use_coeff`	If TRUE both PDP function and its approximation is scaled with corresponding surrogate model coefficient.

Examples

```
library(randomForest)
set.seed(1)
data <- iris
# regression model
iris.rf <- randomForest(Petal.Width ~  Sepal.Length + Petal.Length + Species, data = data)
iris.xs <- xspline(iris.rf)
# plot Sepal.Length transition
plot_variable_transition(iris.xs, "Sepal.Length")
# plot Species transition
plot_variable_transition(iris.xs, "Species")
# plot all transitions
plot_variable_transition(iris.xs)
# plot Sepal.Length transition, its derivative and data points
plot_variable_transition(iris.xs, "Sepal.Length", data = data, plot_data = TRUE, plot_deriv = TRUE)
```

predict.xspliner *Predict xspliner method*

Description

Predict xspliner method

Usage

```
## S3 method for class 'xspliner'
predict(object, newdata, ...)
```

Arguments

object	Object of class 'xspliner'.
newdata	Data that should be prediction based on.
...	Another arguments passed into predict.glm method.

print.xspliner *Print method for xspliner object*

Description

Print method for xspliner object

Usage

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

Arguments

x	xspliner object
predictor	predictor for xspliner model formula
...	Another arguments passed into model specific print method.

stats *Statistics used for better linear model selection*

Description

Used as `compare_stat` parameter in `xspline` method. Each function has attribute "higher-better". If "higher-better" is TRUE then model with higher statistic value is treated as better one.

Usage

```
aic(glm_model)

hoslem(glm_model)
```

Arguments

`glm_model`	Linear model - `glm` function output.

summary.xspliner *Summary method for xspliner object*

Description

Summary method for xspliner object

Usage

```
## S3 method for class 'xspliner'
summary(object, predictor, ..., model = NULL,
  newdata = NULL, prediction_funs = list(function(object, newdata)
  predict(object, newdata)), env = parent.frame())
```

Arguments

`object`	xspliner object
`predictor`	predictor for xspliner model formula
`...`	Another arguments passed into model specific method.
`model`	Original black box model. Providing enables models comparison. See details.
`newdata`	Data used for models comparison. By default training data used for black box build.
`prediction_funs`	List of prediction functions for surrogate and black box model. For classification problem, different statistics are displayed based on predictions type. See details section for more info.
`env`	Environment in which newdata is stored (if not provided as parameter).

Details

The summary output depends strictly on data provided to it.

Standard output for providing only xspliner model (object parameter) return default `glm::summary` output.

Providing both xspliner model and predictor returns summary details for selecter variable. The following points decribe the rules:

- When variable was quantitative and transformed with fitted spline, the output contain approximation details.
- When variable was qualitative and transformed, factor matching is displayed.
- When variable was not transformed, glm::summary output is displayed for the model.

If both object parameter and model (original black box) was provided, the summary displays comparison of original and surrogate model. The following points decribe the rules (y_s and y_o are predictions of surrogate and original model respectively on provided dataset). When comparing statistic is close to 1, this means surrogate model is similiar to black box one (according to this statistic).

For regression models:

- 1 - Maximum predictions normed-difference

$$1 - \frac{\max_{i=1}^{n} |y_s^{(i)} - y_o^{(i)}|}{\max_{i=1}^{n} y_o^{(i)} - \min_{i=1}^{n} y_o^{(i)}}$$

- R^2 (https://christophm.github.io/interpretable-ml-book/global.html#theory-4)

$$1 - \frac{\sum_{i=1}^{n}(y_s^{(i)} - y_o^{(i)})^2}{\sum_{i=1}^{n}(y_o^{(i)} - \overline{y_o})^2}$$

- Mean square errors for each model.

For classification models the result depends on prediction type. When predictions are classified levels:

- Mean predictions similarity

$$\frac{1}{n}\sum_{i=1}^{n} I_{y_s^{(i)}=y_o^{(i)}}$$

- Accuracies for each models.

When predictions are response probabilities:

- R^2 as for regression model.
- 1 - Maximum ROC difference

$$1 - \max_{t \in T} ||ROC_o(t) - ROC_s(t)||_2$$

  Calculates maximum of euclidean distances between ROC points for specified thresholds set T. In this imlplementation T is union of breakpoints for each ROC curve.
- 1 - Mean ROC difference Above version using mean instead of max measure.

Examples

```
library(randomForest)
set.seed(1)
data <- iris
# regression model
iris.rf <- randomForest(Petal.Width ~  Sepal.Length + Petal.Length + Species, data = data)
iris.xs <- xspline(iris.rf)
# Summary of quantitative variable transition
summary(iris.xs, "Sepal.Length")
# Summary of qualitative variable transition
summary(iris.xs, "Species")
# Comparing surrogate with original model (regression)
summary(iris.xs, model = iris.rf, newdata = data)

# Classification model
```

---

transition *Extract variable transformation from xspliner*

Description

Extract variable transformation from xspliner

Usage

```
transition(model, ...)

## S3 method for class 'xspliner'
transition(model, predictor, type = "function", ...)
```

Arguments

model	xspliner model
...	Other parameters passed to method. Currently not used.
predictor	variable name for which transformation should be extracted
type	If 'function' then transformation function is extracted. For 'base' there is sourced object on which transformation was built - in case of quantitative variable GAM model, for qualitative factorMerger.

xf_opts_default *Default parameters for transition methods*

Description

While constructing formula interpreted by xspliner package, some parameters may be specified within xs(..) or xf(..) symbols. Below are default parameters. See details in `vignette("xspliner")`

Usage

```
xf_opts_default

xs_opts_default
```

Format

An object of class `list` of length 2.

xspline *Builds predictive model based GLM.*

Description

The method provides main functionality on building GLM models with automatic variables transformation. The transformations are based on specified single variable responses for selected black-box model. See details in `vignette("xspliner")`.

Usage

```
xspline(object, ...)

model_surrogate_xspliner(object, ...)

## Default S3 method:
xspline(object, lhs = NULL, response = NULL,
  predictors = NULL, data = NULL, form = "additive", bare = NULL,
  env = parent.frame(), ...)

## S3 method for class 'formula'
xspline(object, model, data = NULL,
  consider = "specials", env = parent.frame(), ...)

## S3 method for class 'explainer'
xspline(object, env = parent.frame(), ...)
```

Arguments

object	Predictive model, formula or explainer (see DALEX) object.
...	Other arguments passed to xspline methods or build_xspliner.
lhs	Left-hand side of model formula. Can be transformed response.
response	Name of response variable of model.
predictors	Predictor values that should be used in final model.
data	Training data of model.
form	Can be 'additive' (default) or 'multiplicative'. Specifies formula form in final model.
bare	Variable names that mustn't be transformed in final model.
env	Environment in which optional variables passed into parameters are stored. variables transformation. See vignette("xspliner") for details.
model	When object is formula - predictive model. Basic model used for extracting predictors transformation.
consider	One of c("specials","all"). If "specials", only components with xs or xf call are considered in transition.

Details

model_surrogate_xspliner is a wrapper of xspline method to assure consistency with https://github.com/ModelOriented/DrWh
tools

Value

GLM object of class 'xspliner'.

Examples

```
# preparing blackbox model
library(randomForest)
rf_iris <- randomForest(
  Petal.Width ~  Sepal.Length + Petal.Length + Species,
  data = iris)

# formula based xspliner
xs_iris <- xspline(
  Petal.Width ~ xs(Sepal.Length) + xs(Petal.Length) + xf(Species),
  model = rf_iris)
summary(xs_iris)
plot(xs_iris, "Sepal.Length")

# passing just the model
xs_iris <- xspline(rf_iris)
summary(xs_iris)
plot(xs_iris, "Sepal.Length")

# using DALEX
```

```
library(DALEX)
xs_iris_explainer <- explain(rf_iris)
xs_iris <- xspline(rf_iris)
summary(xs_iris)
plot(xs_iris, "Sepal.Length")
```

Index