Package 'ciu'

October 25, 2020

Type Package

Title Contextual Importance and Utility

Version 0.1.0

Author Kary Främling

Maintainer Kary Främling <Kary.Framling@umu.se>

Description R implementation of the Contextual Importance and Utility (CIU) concepts for Explainable AI (XAI). A recent description of CIU can be found in e.g. Framling (2020) <arXiv:2009.13996>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports graphics, stats, Rcpp, grDevices, ggplot2

Suggests MASS, caret

RoxygenNote 7.1.1

Roxygen list(markdown = TRUE)

NeedsCompilation no

R topics documented:

ciu-package *ciu: Contextual Importance and Utility*

Description

R implementation of the Contextual Importance and Utility (CIU) concepts for Explainable AI (XAI). A recent description of CIU can be found in e.g. Framling (2020) <arXiv:2009.13996>.

Details

This package implements the Contextual Importance and Utility (CIU) concepts for Explainable AI (XAI). CIU allows explaining outputs values of any regression or classification systems, no matter if it is a "black-box" or a "white-box" AI, or anything between black and white. CIU is entirely model-agnostic. Contrary to most (all?) other XAI methods, CIU provides explanations directly based on the observed input-output behavior without building an intermediate "interpretable" model for doing it.

CIU was developed by Kary Främling in his PhD thesis, which was presented in 1996 (in French). CIU was first presented in 1995 at the International Conference on Artificial Neural Networks (ICANN).

The ciu package supports models from `caret` and at least `lda` natively, but can easily be made to work with any model.

Main functions:

Use of `ciu` starts by calling the function ciu.new that returns an object of class `CIU`. If the returned object is stored in a variable called `ciu`, then different methods can be called as `ciu$explain()`, `ciu$barplot.ciu()` etc. for obtaining explanations in different forms.

`ciu` is implemented using an "old style" (?) R object orientation. However, it provides object-oriented encapsulation of variables and methods of the `CIU` object, which presumably helps to avoid name conflicts with other packages or user code.

References

Främling, K. *Modélisation et apprentissage des préférences par réseaux de neurones pour l'aide à la décision multicritère*. 1996, https://tel.archives-ouvertes.fr/tel-00825854/document (title translation in English: *Learning and Explaining Preferences with Neural Networks for Multiple Criteria Decision Making*)

barplot.ciu *Barplot CIU explanation for specific instance*

Description

Create a barplot showing CI as the length of the bar and CU on color scale from red to green, via yellow, for the given inputs and the given output. First get a `CIU` object by calling ciu.new as e.g. `ciu <-ciu.new(...)`, then call as `ciu.res <-ciu$barplot.ciu(...)`.

Arguments

`instance`	the current input values for the instance to explain/plot.
`ind.inputs`	vector of indices for the inputs to be included in the plot. If NULL then all inputs will be included.
`ind.output`	index of output to be explained.
`in.min.max.limits`	matrix with one row per output and two columns, where the first column indicates the minimal value and the second column the maximal value for that output. ONLY NEEDED if they were not provided to "new" or if they need to be different for this instance.
`n.samples`	how many random instances to use for estimating CI and CU.
`neutral.CU`	indicates when the Contextual Utility is considered to be "negative". The default value of 0.5 seems quite logical for most cases.
`show.input.values`	include input values after input labels or not. Default is TRUE.
`concepts.to.explain`	list of concepts to use in the plot, as defined by vocabulary provided as argument to ciu.new. if "ind.inputs=NULL", then use "concepts.to.explain" instead. If both are NULL, then use all inputs.
`target.concept`	if provided, then calculate CIU of inputs `ind.inputs.to.explain` relative to the given concept rather than relative to the actual output(s). See `explain()`.
`target.ciu`	See `explain()`.
`color.ramp.below.neutral`	color ramp function as returned by function colorRamp(). Default color ramp is from red3 to yellow.
`color.ramp.above.neutral`	color ramp function as returned by function colorRamp(). Default color ramp is from yellow to darkgreen.
`sort`	NULL, "CI" or "CU". No sorting by default, other options are sorting by CI or CU.
`decreasing`	set to TRUE for decreasing sort (see arguments of `sort()`).
`main, xlab, xlim, ...`	usual plot parameters, possible to override the default ones provided here if needed.

Value

"void", i.e. whatever happens to be result of last instruction.

Author(s)

Kary Främling

`ciu.new` *Create CIU object*

Description

Sets up a CIU object with the given parameters. CIU objects have "public" and "private" methods. A CIU object is actually a list whose elements are the public functions (methods).

Usage

```
ciu.new(
  bb,
  formula = NULL,
  data = NULL,
  in.min.max.limits = NULL,
  abs.min.max = NULL,
  input.names = NULL,
  output.names = NULL,
  predict.function = NULL,
  vocabulary = NULL
)
```

Arguments

`bb`	Model/"black-box" object. At least all `caret` models, the `lda` model from MASS, and the `lm` model are supported. Otherwise, the prediction function to be used can be gives as value of the `predict.function` parameter. A more powerful way is to inherit from FunctionApproximator class and implement an "eval" method.
`formula`	Formula that describes input versus output values. Only to be used together with `data` parameter.
`data`	The training data used for training the model. If this parameter is provided, a `formula` MUST be given also. `ciu.new` attempts to infer the other parameters from `data` and `formula`. i.e. `in.min.max.limits`, `abs.min.max`, `input.names` and `output.names`. If those parameters are provided, then they override the inferred ones.
`in.min.max.limits`	matrix with one row per output and two columns, where the first column indicates the minimal value and the second column the maximal value for that input.
`abs.min.max`	data.frame or matrix of min-max values of outputs, one row per output, two columns (min, max).
`input.names`	labels of inputs.
`output.names`	labels of outputs.
`predict.function`	can be supplied if a model that is not supported by ciu should be used. As an example, this is the function for lda:

```
o.predict.function <- function(model, inputs) {
    pred <- predict(model,inputs)
        return(pred$posterior)
}
```

vocabulary — list of labels/concepts to be used when producing explanations and what combination of inputs they correspond to. Example of two intermediate concepts and a higher-level one that combines them: `list(intermediate.concept1=c(1,2,3),intermediate.co`

Value

Object of class `CIU`.

Author(s)

Kary Främling

See Also

`explain` method

`barplot.ciu` method

Examples

```
# Explaining the classification of an Iris instance with lda model.
# We use a versicolor (instance 100).
library(MASS)
test.ind <- 100
iris_test <- iris[test.ind, 1:4]
iris_train <- iris[-test.ind, 1:4]
iris_lab <- iris[[5]][-test.ind]
model <- lda(iris_train, iris_lab)

# Create CIU object
ciu <- ciu.new(model, Species~., iris)

# This can be used with explain method for getting CIU values
# of one or several inputs. Here we get CIU for all three outputs
# with input feature "Petal.Length" that happens to be the most important.
ciu$explain(iris_test, 1)

# It is, however, more convenient to use one of the graphical visualisations.
# Here's one using ggplot.
ciu$ggplot.col.ciu(iris_test)

# LDA creates very sharp class limits, which can also be seen in the CIU
# explanation. We can study what the underlying model looks like using
# plot.ciu and plot.ciu.3D methods. Here is a 3D plot for all three classes
# as a function of Petal Length&Width. Iris #100 (shown as the red dot)
# is on the ridge of the "versicolor" class, which is quite narrow for
# Petal Length&Width.
par(mfrow=c(1,3))
ciu$plot.ciu.3D(iris_test,c(3,4),1,main=levels(iris$Species)[1],)
ciu$plot.ciu.3D(iris_test,c(3,4),2,main=levels(iris$Species)[2])
ciu$plot.ciu.3D(iris_test,c(3,4),3,main=levels(iris$Species)[3])
par(mfrow=c(1,1))
```

```
# Same thing with a regression task, the Boston Housing data set. Instance
# #370 has the highest valuation (50k$). Model is gbm, which performs
# decently here. Plotting with "standard" bar plot this time.
library(caret)
gbm <- train(medv ~ ., Boston, method="gbm", trControl=trainControl(method="cv", number=10))
ciu <- ciu.new(gbm, medv~., Boston)
ciu$barplot.ciu(Boston[370,1:13])

# Same but sort by CI.
ciu$barplot.ciu(Boston[370,1:13], sort = "CI")
```

`ciu.relative` *Calculate relative CIU of a sub-concept/input relative to an intermediate concept (or output).*

Description

Calculate relative CIU of a sub-concept/input relative to an intermediate concept (or output). The parameters must be of class "ciu.result" or a data.frame with compatible columns.

Usage

```
ciu.relative(sub.ciu.result, sup.ciu.result)
```

Arguments

`sub.ciu.result`	ciu.result object of sub-concept/input.
`sup.ciu.result`	ciu.result object of intermediate concept/output.

`ciu.result.new` *CIU result object*

Description

Create object of class `ciu.result`, which stores results of CIU calculations. The `explain()` method returns a `ciu.result` object.

Usage

```
ciu.result.new(ci, cu, cmin, cmax, outval)
```

Arguments

`ci`	vector of CI values, one per output
`cu`	vector of CU values, one per output
`cmin`	vector of cmin values, one per output
`cmax`	vector of cmax values, one per output
`outval`	vector of black-box output values, one per output

Value

An object of class ciu.result, which is a data.frame with (at least) five columns:

- CI values: one row per output of the black-box model
- CU values: one row per output of the black-box model
- cmin values: one row per output of the black-box model
- cmax values: one row per output of the black-box model
- outval values: one row per output of the black-box model

Author(s)

Kary Främling

`explain` *Calculate CIU for specific instance*

Description

Calculate Contextual Importance (CI) and Contextual Utility (CU) for an instance (Context) using the given "black-box" model. First get a `CIU` object by calling `ciu.new` as e.g. `ciu <-ciu.new(...)`, then call as `ciu.res <-ciu$explain(...)`.

Arguments

`instance`	the current input values for the instance to explain. Should be a data.frame even though a vector or matrix might work too if input names and other needed metadata can be deduced from the dataset or other parameters given to `ciu.new`.
`ind.inputs.to.explain`	vector of indices for the inputs to be explained, i.e. for which CIU should be calculated. If NULL, then all inputs will be included.
`in.min.max.limits`	data.frame or matrix with one row per output and two columns, where the first column indicates the minimal value and the second column the maximal value for that output. ONLY NEEDED HERE IF not given as parameter to ciu.new or if the limits are different for this specific instance than the default ones.
`n.samples`	how many instances to generate for estimating CI and CU. For inputs of type factor, all possible combinations of input values is generated, so this parameter only influences how many instances are (at least) generated for continuous-valued inputs.
`target.concept`	if provided, then calculate CIU of inputs `ind.inputs.to.explain` relative to the given concept rather than relative to the actual output(s). `ind.inputs.to.explain` should normally be a subset (or all) of the inputs that `target.concept` consists of, even though that not required by the CIU calculation. If a "target.ciu" is provided, then the "target.concept" doesn't have to be included in the `vocabulary` gives as parameter to ciu.new (at least for the moment).
`target.ciu`	`ciu.result` object previously calculated for `target.concept`. Default value is NULL. If a `target.concept` is provided but `target.ciu=NULL`, then `target.ciu` is estimated by a call to "explain" with the `n.samples` value given as a parameter to this call. It may be useful to provide `target.ciu` if it should be estimated using some other (typically greater) value for `n.samples` than the default one, or if it has already been calculated for some reason.

Value

A `ciu.result` object (see ciu.result.new)

Author(s)

Kary Främling

`ggplot.col.ciu` *CIU explanation as* ggplot col.

Description

CIU explanation as ggplot col.

Index