Package 'fairadapt'

October 9, 2023

Title Fair Data Adaptation with Quantile Preservation

Description An implementation of the fair data adaptation with quantile preservation described in Plecko & Meinshausen (2019) <arXiv:1911.06685>. The adaptation procedure uses the specified causal graph to pre-process the given training and testing data in such a way to remove the bias caused by the protected attribute. The procedure uses tree ensembles for quantile regression.

Version 0.2.7

License GPL (>= 3)

Encoding UTF-8

Language en-US

LazyData true

URL https://github.com/dplecko/fairadapt

BugReports https://github.com/dplecko/fairadapt/issues

Depends R (>= 3.5.0)

Imports ranger (>= 0.13.1), assertthat, quantreg, qrnn, igraph, ggplot2, cowplot, scales

Suggests testthat (>= 3.0.3), knitr, rmarkdown, rticles, mvtnorm, magick, ggraph, pdftools, microbenchmark, xtable

RoxygenNote 7.2.3

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Drago Plecko [aut, cre], Nicolas Bennett [aut]

Maintainer Drago Plecko <drago.plecko@stat.math.ethz.ch>

Repository CRAN

Date/Publication 2023-10-09 18:30:05 UTC

R topics documented:

`adaptedData` *Convenience function for returning adapted data*

Description

Convenience function for returning adapted data

Usage

```
adaptedData(x, train = TRUE)

## S3 method for class 'fairadapt'
adaptedData(x, train = TRUE)

## S3 method for class 'fairadaptBoot'
adaptedData(x, train = TRUE)
```

Arguments

`x`	Object of class `fairadapt` or `fairadaptBoot`, a result of an adaptation procedure.
`train`	A logical indicating whether train data should be returned. Defaults to `TRUE`. If `FALSE`, test data is returned.

Value

Either a `data.frame` when called on an `fairadapt` object, or a `list` of `data.frames` with the adapted data of length `n.boot`, when called on a `fairadaptBoot` object.

compas *COMPAS dataset.*

Description

A real dataset from Broward County, Florida. Contains information on individuals released on parole, and whether they reoffended within two years.

Usage

```
compas
```

Format

A data frame with 1,000 rows and 9 variables:

sex sex of the individual

age age, measured in years

race race, binary with values Non-White and White

juv_fel_count count of juvenile felonies

juv_misd_count count of juvenile misdemeanors

juv_other_count count of other juvenile offenses

priors_count count of prior offenses

c_charge_degree degree of charge, with two values, F (felony) and M (misdemeanor)

two_year_recid a logical TRUE/FALSE indicator of recidivism within two years after parole start

computeQuants *Compute Quantiles generic for the Quantile Learning step.*

Description

Compute Quantiles generic for the Quantile Learning step.

Usage

```
computeQuants(x, data, newdata, ind, ...)
```

Arguments

x	Object with an associated computeQuants() method, to be used for inferring quantiles.
data	data.frame containing samples used in the quantile regression.
newdata	data.frame containing counterfactual values for which the quantiles need to be inferred.
ind	A logical vector of length nrow(data), indicating which samples have the baseline value of the protected attribute.
...	Additional arguments to be passed down to respective method functions.

Value

A vector of counterfactual values corresponding to newdata.

---

fairadapt *Fairadapt*

Description

Implementation of fair data adaptation with quantile preservation (Plecko & Meinshausen 2019). Uses only plain R.

Usage

```
fairadapt(
  formula,
  prot.attr,
  adj.mat,
  train.data,
  test.data = NULL,
  cfd.mat = NULL,
  top.ord = NULL,
  res.vars = NULL,
  quant.method = rangerQuants,
  visualize.graph = FALSE,
  eval.qfit = NULL,
  ...
)
```

Arguments

formula	Object of class formula describing the response and the covariates.
prot.attr	A value of class character describing the binary protected attribute. Must be one of the entries of colnames(adj.mat).

`adj.mat` Matrix of class `matrix` encoding the relationships in the causal graph. `M[i,j] == 1L` implies the existence of an edge from node i to node j. Must include all the variables appearing in the formula object. When the `adj.mat` argument is set to `NULL`, then the `top.ord` argument has to be supplied.

`train.data, test.data` Training data & testing data, both of class `data.frame`. Test data is by default `NULL`.

`cfd.mat` Symmetric matrix of class `matrix` encoding the bidirected edges in the causal graph. `M[i,j] == M[j, i] == 1L` implies the existence of a bidirected edge between nodes i and j. Must include all the variables appearing in the formula object.

`top.ord` A vector of class `character` describing the topological ordering of the causal graph. Default value is `NULL`, but this argument must be supplied if `adj.mat` is not specified. Also must include all the variables appearing in the formula object.

`res.vars` A vector of class `character` listing all the resolving variables, which should not be changed by the adaption procedure. Default value is `NULL`, corresponding to no resolving variables. Resolving variables should be a subset of the descendants of the protected attribute.

`quant.method` A function choosing the method used for quantile regression. Default value is `rangerQuants` (using random forest quantile regression). Other implemented options are `linearQuants` and `mcqrnnQuants`. A custom function can be supplied by the user here, and the associated method for the S3 generic `computeQuants` needs to be added.

`visualize.graph` A `logical` indicating whether the causal graph should be plotted upon calling the `fairadapt()` function. Default value is `FALSE`.

`eval.qfit` Argument indicating whether the quality of the quantile regression fit should be computed using cross-validation. Default value is `NULL`, but whenever a positive integer value is specified, then it is interpreted as the number of folds used in the cross-validation procedure.

`...` Additional arguments forwarded to the function passed as `quant.method`.

Details

The procedure takes the training and testing data as an input, together with the causal graph given by an adjacency matrix and the list of resolving variables, which should be kept fixed during the adaptation procedure. The procedure then calculates a fair representation of the data, after which any classification method can be used. There are, however, several valid training options yielding fair predictions, and the best of them can be chosen with cross-validation. For more details we refer the user to the original paper. Most of the running time is due to the quantile regression step using the ranger package.

Value

An object of class `fairadapt`, containing the original and adapted training and testing data, together with the causal graph and some additional meta-information.

References

Plecko, D. & Meinshausen, N. (2019). Fair Data Adaptation with Quantile Preservation

Examples

```
n_samp <- 200
uni_dim <- c(       "gender", "edu", "test", "score")
uni_adj <- matrix(c(       0,     1,      1,       0,
                           0,     0,      1,       1,
                           0,     0,      0,       1,
                           0,     0,      0,       0),
                  ncol = length(uni_dim),
                  dimnames = rep(list(uni_dim), 2),
                  byrow = TRUE)

uni_ada <- fairadapt(score ~ .,
  train.data = head(uni_admission, n = n_samp),
  test.data = tail(uni_admission, n = n_samp),
  adj.mat = uni_adj,
  prot.attr = "gender"
)

uni_ada
```

`fairadaptBoot` *Fairadapt Boostrap wrapper*

Description

The `fairadapt()` function performs data adaptation, but does so only once. Sometimes, it might be desirable to repeat this process, in order to be able to make uncertainty estimates about the data adaptation that is performed. The wrapper function `fairadaptBoot()` enables the user to do so, by performing the `fairadapt()` procedure multiple times, and keeping in memory the important multiple data transformations. For a worked example of how to use `fairadaptBoot()` for uncertainty quantification, see the `fairadapt` vignette.

Usage

```
fairadaptBoot(
  formula,
  prot.attr,
  adj.mat,
  train.data,
  test.data = NULL,
  cfd.mat = NULL,
  top.ord = NULL,
  res.vars = NULL,
```

```
  quant.method = rangerQuants,
  keep.object = FALSE,
  n.boot = 100,
  rand.mode = c("finsamp", "quant", "both"),
  test.seed = 2022,
  ...
)
```

Arguments

formula	Object of class formula describing the response and the covariates.
prot.attr	A value of class character describing the binary protected attribute. Must be one of the entries of colnames(adj.mat).
adj.mat	Matrix of class matrix encoding the relationships in the causal graph. M[i,j] == 1L implies the existence of an edge from node i to node j. Must include all the variables appearing in the formula object. When the adj.mat argument is set to NULL, then the top.ord argument has to be supplied.
train.data, test.data	Training data & testing data, both of class data.frame. Test data is by default NULL.
cfd.mat	Symmetric matrix of class matrix encoding the bidirected edges in the causal graph. M[i,j] == M[j, i] == 1L implies the existence of a bidirected edge between nodes i and j. Must include all the variables appearing in the formula object.
top.ord	A vector of class character describing the topological ordering of the causal graph. Default value is NULL, but this argument must be supplied if adj.mat is not specified. Also must include all the variables appearing in the formula object.
res.vars	A vector of class character listing all the resolving variables, which should not be changed by the adaption procedure. Default value is NULL, corresponding to no resolving variables. Resolving variables should be a subset of the descendants of the protected attribute.
quant.method	A function choosing the method used for quantile regression. Default value is rangerQuants (using random forest quantile regression). Other implemented options are linearQuants and mcqrnnQuants. A custom function can be supplied by the user here, and the associated method for the S3 generic computeQuants needs to be added.
keep.object	a logical scalar, indicating whether all the fairadapt S3 objects built in bootstrap repetitions should be saved.
n.boot	An integer corresponding to the umber of bootstrap iterations.
rand.mode	A string, taking values "finsamp", "quant" or "both", corresponding to considering finite sample uncertainty, quantile uncertainty, or both.
test.seed	a seed for the randomness in breaking quantiles for the discrete variables. This argument is only relevant when rand.mode equals "quant" or "both" (otherwise ignored).
...	Additional arguments forwarded to the function passed as quant.method.

Value

An object of class `fairadaptBoot`, containing the original and adapted training and testing data, together with the causal graph and some additional meta-information.

References

Plecko, D. & Meinshausen, N. (2019). Fair Data Adaptation with Quantile Preservation

Examples

```
n_samp <- 200
uni_dim <- c(       "gender", "edu", "test", "score")
uni_adj <- matrix(c(       0,     1,      1,       0,
                           0,     0,      1,       1,
                           0,     0,      0,       1,
                           0,     0,      0,       0),
                  ncol = length(uni_dim),
                  dimnames = rep(list(uni_dim), 2),
                  byrow = TRUE)

uni_ada <- fairadaptBoot(score ~ .,
  train.data = head(uni_admission, n = n_samp),
  test.data = tail(uni_admission, n = n_samp),
  adj.mat = uni_adj,
  prot.attr = "gender",
  n.boot = 5
)

uni_ada
```

`fairTwins` *Fair Twin Inspection convenience function.*

Description

Fair Twin Inspection convenience function.

Usage

```
fairTwins(x, train.id = seq_len(nrow(x$train)), test.id = NULL, cols = NULL)
```

Arguments

`x`	Object of class `fairadapt`, a result of an adaptation procedure.
`train.id`	A vector of indices specifying which rows of the training data should be displayed.
`test.id`	A vector of indices specifying which rows of the test data should be displayed.
`cols`	A `character` vector, subset of `names(train.data)`, which specifies which subset of columns is to be displayed in the result.

Value

A `data.frame`, containing the original and adapted values of the requested individuals. Adapted columns have `_adapted` appended to their original name.

Examples

```
n_samp <- 200
uni_dim <- c(       "gender", "edu", "test", "score")
uni_adj <- matrix(c(       0,     1,      1,       0,
                           0,     0,      1,       1,
                           0,     0,      0,       1,
                           0,     0,      0,       0),
                  ncol = length(uni_dim),
                  dimnames = rep(list(uni_dim), 2),
                  byrow = TRUE)

uni_ada <- fairadapt(score ~ .,
  train.data = head(uni_admission, n = n_samp),
  test.data = tail(uni_admission, n = n_samp),
  adj.mat = uni_adj,
  prot.attr = "gender"
)

fairTwins(uni_ada, train.id = 1:5)
```

`gov_census` *Census information of US government employees.*

Description

The dataset contains various demographic, education and work information of the employees of the US government. The data is taken from the 2018 US Census data.

Usage

```
gov_census
```

Format

A data frame with 204,309 rows and 17 variables:

sex gender of the employee

age employee age in years

race race of the employee

hispanic_origin indicator of hispanic origin

citizenship citizenship of the employee

nativity indicator of nativity to the US

marital marital status

family_size size of the employee's family

children number of children of the employee

education_level education level measured in years

english_level

salary yearly salary in US dollars

hours_worked hours worked every week

weeks_worked weeks worked in the given year

occupation occupation classification

industry industry classification

economic_region economic region where the person is employed in the US

Source

https://www.census.gov/programs-surveys/acs/microdata/documentation.html

---

`graphModel` *Obtaining the graphical causal model (GCM)*

Description

Obtaining the graphical causal model (GCM)

Usage

```
graphModel(adj.mat, cfd.mat = NULL, res.vars = NULL)
```

Arguments

`adj.mat`	Matrix of class `matrix` encoding the relationships in the causal graph. `M[i,j] == 1L` implies the existence of an edge from node i to node j.
`cfd.mat`	Symmetric matrix of class `matrix` encoding the bidirected edges in the causal graph. `M[i,j] == M[j, i] == 1L` implies the existence of a bidirected edge between nodes i and j.
`res.vars`	A vector of class `character` listing all the resolving variables, which should not be changed by the adaption procedure. Default value is `NULL`, corresponding to no resolving variables. Resolving variables should be a subset of `colnames(adj.mat)`. Resolving variables are marked with a different color in the output.

Value

An object of class `igraph`, containing the causal graphical, with directed and bidirected edges.

Examples

```
adj.mat <- cfd.mat <- array(0L, dim = c(3, 3))
colnames(adj.mat) <- rownames(adj.mat) <-
  colnames(cfd.mat) <- rownames(cfd.mat) <- c("A", "X", "Y")

adj.mat["A", "X"] <- adj.mat["X", "Y"] <-
  cfd.mat["X", "Y"] <- cfd.mat["Y", "X"] <- 1L

gcm <- graphModel(adj.mat, cfd.mat, res.vars = "X")
```

---

predict.fairadapt *Prediction function for new data from a saved* fairadapt *object.*

Description

Prediction function for new data from a saved fairadapt object.

Usage

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

Arguments

object	Object of class fairadapt, a result of an adaptation procedure.
newdata	A data.frame containing the new data.
...	Additional arguments forwarded to computeQuants().

Details

The newdata argument should be compatible with adapt.test argument that was used when constructing the fairadapt object. In particular, newdata should contain column names that appear in the formula argument that was used when calling fairadapt() (apart from the outcome variable on the LHS of the formula).

Value

A data.frame containing the adapted version of the new data.

Examples

```
n_samp <- 200
uni_dim <- c(       "gender", "edu", "test", "score")
uni_adj <- matrix(c(       0,     1,      1,       0,
                           0,     0,      1,       1,
                           0,     0,      0,       1,
                           0,     0,      0,       0),
```

```
                  ncol = length(uni_dim),
                  dimnames = rep(list(uni_dim), 2),
                  byrow = TRUE)

uni_ada <- fairadapt(score ~ .,
  train.data = head(uni_admission, n = n_samp),
  adj.mat = uni_adj,
  prot.attr = "gender"
)

predict(object = uni_ada, newdata = tail(uni_admission, n = n_samp))
```

`predict.fairadaptBoot` *Prediction function for new data from a saved* `fairadaptBoot` *object.*

Description

Prediction function for new data from a saved `fairadaptBoot` object.

Usage

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

Arguments

`object`	Object of class `fairadapt`, a result of an adaptation procedure.
`newdata`	A `data.frame` containing the new data.
`...`	Additional arguments forwarded to `computeQuants()`.

Details

The `newdata` argument should be compatible with `adapt.test` argument that was used when constructing the `fairadaptBoot` object. In particular, `newdata` should contain column names that appear in the `formula` argument that was used when calling `fairadaptBoot()` (apart from the outcome variable on the LHS of the formula).

Value

A `data.frame` containing the adapted version of the new data.

Examples

```
n_samp <- 200
uni_dim <- c(       "gender", "edu", "test", "score")
uni_adj <- matrix(c(       0,     1,      1,       0,
                           0,     0,      1,       1,
                           0,     0,      0,       1,
```

```
                        0,     0,      0,       0),
                  ncol = length(uni_dim),
                  dimnames = rep(list(uni_dim), 2),
                  byrow = TRUE)

uni_ada_boot <- fairadaptBoot(score ~ .,
  train.data = head(uni_admission, n = n_samp),
  adj.mat = uni_adj,
  prot.attr = "gender",
  n.boot = 5,
  keep.object = TRUE
)

predict(object = uni_ada_boot, newdata = tail(uni_admission, n = n_samp))
```

`quantFit` *Quality of quantile fit statistics.*

Description

Quality of quantile fit statistics.

Usage

```
quantFit(x, ...)
```

Arguments

`x`	Object of class `fairadapt`, a result of an adaptation procedure.
`...`	Ignored in this case.

Value

A `numeric` vector, containing the average empirical loss for the 25%, 50% and 75% quantile loss functions, for each variable.

Examples

```
n_samp <- 200
uni_dim <- c(       "gender", "edu", "test", "score")
uni_adj <- matrix(c(       0,     1,      1,       0,
                           0,     0,      1,       1,
                           0,     0,      0,       1,
                           0,     0,      0,       0),
                  ncol = length(uni_dim),
                  dimnames = rep(list(uni_dim), 2),
                  byrow = TRUE)

uni_ada <- fairadapt(score ~ .,
```

```
  train.data = head(uni_admission, n = n_samp),
  test.data = tail(uni_admission, n = n_samp),
  adj.mat = uni_adj,
  prot.attr = "gender",
  eval.qfit = 3L
)

quantFit(uni_ada)
```

`rangerQuants` *Quantile Engine Constructor for the Quantile Learning step.*

Description

There are several methods that can be used for the quantile learning step in the `fairadapt` package. Each of the methods needs a specific constructor. The constructor is a function that takes the data (with some additional meta-information) and returns an object on which the `computeQuants()` generic can be called.

Usage

```
rangerQuants(data, A.root, ind, min.node.size = 20, ...)

linearQuants(
  data,
  A.root,
  ind,
  tau = c(0.001, seq(0.005, 0.995, by = 0.01), 0.999),
  ...
)

mcqrnnQuants(
  data,
  A.root,
  ind,
  tau = seq(0.005, 0.995, by = 0.01),
  iter.max = 500,
  ...
)
```

Arguments

`data`	A `data.frame` with data to be used for quantile regression.
`A.root`	A `logical(1L)` indicating whether the protected attribute `A` is a root node of the causal graph. Used for splitting the quantile regression.
`ind`	A `logical` vector of length `nrow(data)`, indicating which samples have the baseline value of the protected attribute.

min.node.size	Forwarded to ranger::ranger().
...	Forwarded to further methods.
tau	Forwarded to quantreg::rq() or qrnn::mcqrnn.fit().
iter.max	Forwarded to qrnn::mcqrnn.fit().

Details

Within the package, there are 3 different methods implemented, which use quantile regressors based on linear models, random forests and neural networks. However, there is additional flexibility and the user can provide her/his own quantile method. For this, the user needs to write (i) the constructor which returns an S3 classed object (see examples below); (ii) a method for the computeQuants() generic for the S3 class returned in (i).

The rangerQuants() function uses random forests (ranger package) for quantile regression.

The linearQuants() function uses linear quantile regression (quantreg package) for the Quantile Learning step.

The mcqrnnQuants() function uses monotone quantile regression neural networks (mcqrnn package) in the Quantile Learning step.

Value

A ranger or a rangersplit S3 object, depending on the value of the A.root argument, for rangerQuants().

A rqs or a quantregsplit S3 object, depending on the value of the A.root argument, for linearQuants().

An mcqrnn S3 object for mcqrnnQuants().

---

uni_admission *University admission data of 1,000 students.*

Description

A simulated dataset containing the evaluation of students' abilities.

Usage

```
uni_admission
```

Format

A data frame with 1,000 rows and 4 variables:

gender the gender of the student

edu educational achievement, for instance GPA

test performance on a university admission test

score overall final score measuring the quality of a candidate

`visualizeGraph` *Visualize Graphical Causal Model*

Description

Visualize Graphical Causal Model

Usage

```
visualizeGraph(x, ...)
```

Arguments

`x`	Object of class `fairadapt`, a result of an adaptation procedure.
`...`	Additional arguments passed to the graph plotting function.

Index