# Package ‘fairml’

August 6, 2020

**Type** Package

**Title** Fair Models in Machine Learning

**Version** 0.2

**Date** 2020-07-30

**Depends** R (>= 3.5.0)

**Imports** methods, optiSolve

**Suggests** lattice

**Author** Marco Scutari [aut, cre]

**Maintainer** Marco Scutari <marco.scutari@gmail.com>

**Description** Fair machine learning regression models which take sensitive attributes into account in model estimation. Currently implementing Komiyama et al. (2018) <http://proceedings.mlr.press/v80/komiyama18a/komiyama18a.pdf>.

**License** MIT + file LICENSE

**LazyData** yes

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2020-08-06 14:10:03 UTC

## R topics documented:

## fairml-package *Fair models in machine learning*

### Description

Fair machine learning models: estimation, tuning and prediction.

### Details

**fairml** implements key algorithms for learning machine learning models while enforcing fairness with respect to a set of observed sensitive (or protected) attributes.

Currently **fairml** implements the following algorithms (references below):

- `nclm()`: the non-convex formulation of fair linear regression from Komiyama et al. (2018).

### Author(s)

Marco Scutari
Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA)

Maintainer: Marco Scutari <`marco.scutari@gmail.com`>

### References

Komiyama J, Takeda A, Honda J, Shimao H (2018). "Nonconvex Optimization for Regression with Fairness Constraints". Proceedints of the 35th International Conference on Machine Learning (ICML), PMLR **80**:2737–2746.

## communities.and.crime *iCommunities and Crime Data Set*

### Description

Combined socio-economic data from the 1990 Census, law enforcement data from the 1990 LEMAS survey, and crime data from the 1995 FBI UCR for various communities in the United States.

### Usage

```
data(communities.and.crime)
```

### Format

The data contains 1969 observations and 104 variables. See the UCI Machine Learning Repository for details.

## Note

The data set has been pre-processed as in Komiyama et al. (2018), with the following exceptions:

- the variable `community` has been dropped, as it is non-predictive and contains a sizeable number of missing values;
- the variables `LemasSwornFT`, `LemasSwFTPerPop`, `LemasSwFTFieldOps`, `LemasSwFTFieldPerPop`, `LemasTotalReq`, `LemasTotReqPerPop`, `PolicReqPerOffic`, `PolicPerPop`, `RacialMatchCommPol`, `PctPolicWhite`, `PctPolicBlack`, `PctPolicHisp`, `PctPolicAsian`, `PctPolicMinor`, `OfficAssgnDrugUnits`, `NumKindsDrugsSeiz`, `PolicAveOTWorked`, `PolicCars`, `PolicOperBudg`, `LemasPctPolicOnPatr`, `LemasGangUnitDeploy` and `PolicBudgPerPop` have been dropped because they have more than 80% missing values.

In that paper, `ViolentCrimesPerPop` is the response variable, `racepctblack` and `PctForeignBorn` are the sensitive attributes and the remaining variables are used as predictors.

## References

UCI Machine Learning Repository: http://archive.ics.uci.edu/ml/datasets/communities+and+crime

## Examples

```
data(communities.and.crime)

# short-hand variable names.
cc = communities.and.crime[complete.cases(communities.and.crime), ]
r = cc[, "ViolentCrimesPerPop"]
s = cc[, c("racepctblack", "PctForeignBorn")]
p = cc[, setdiff(names(cc), c("ViolentCrimesPerPop", names(s)))]

m = nclm(response = r, sensitive = s, predictors = p, epsilon = 0.05)

summary(m)
```

# `compas` *Criminal Offenders Screened in Florida*

## Description

A collection of criminal offenders screened in Florida (US) during 2013-14.

## Usage

```
data(compas)
```

## Format

The data contains 5855 observations and the following variables:

- age, a continuous variable containing the age (in years) of the person;
- juv_fel_count, a continuous variable containing the number of juvenile felonies;
- decile_score, a continuous variable, the decile of the COMPAS score;
- juv_misd_count, a continuous variable containing the number of juvenile misdemeanors;
- juv_other_count, a continuous variable containing the number of prior juvenile convictions that are not considered either felonies or misdemeanors;
- v_decile_score, a continuous variable containing the predicted decile of the COMPAS score;
- priors_count, a continuous variable containing the number of prior crimes committed;
- sex, a factor with levels "Female" and "Male";
- two_year_recid, a factor with two levels "Yes" and "No" (if the person has recidivated within two years);
- race, a factor encoding the race of the person;
- c_jail_in, a numeric variable containing the date in which the person entered jail (normalized between 0 and 1);
- c_jail_out, a numeric variable containing the date in which the person was released from jail (normalized between 0 and 1);
- c_offense_date, a numeric variable containing the date the offense was committed;
- screening_date, a numeric variable containing the date in which the person was screened (normalized between 0 and 1);
- in_custody, a numeric variable containing the date in which the person was placed in custody (normalized between 0 and 1);
- out_custody, a numeric variable containing the date in which the person was released from custody (normalized between 0 and 1);

## Note

The data set has been pre-processed as in Komiyama et al. (2018), with the following exceptions:

- the race variable has not been reduced to a binary factor with levels "African-American" and "not African-American";
- the variables type_of_assessment, v_type_of_assessment have been dropped from the analysis because they take the same value for all observations;
- variables like c_jail_in and c_jail_out that encode dates have been jointly rescaled to preserve the temporal ordering of events.

In that paper, two_year_recid is the response variable, sex and race are the sensitive attributes and the remaining variables are used as predictors.

### References

Angwin J, Larson J, Mattu S, Kirchner L (2016). "Machine Bias: Theres Software Used Around the Country to Predict Future Criminals." https://www.propublica.org.

### Examples

```
data(compas)

# convert the response back to a numeric variable.
compas$two_year_recid = as.numeric(compas$two_year_recid) - 1

m = nclm(response = compas[, "two_year_recid"],
      sensitive = compas[, c("sex", "race")],
      predictors = compas[, setdiff(names(compas), c("two_year_recid", "sex", "race"))],
      epsilon = 0.05)

summary(m)
```

## `fairness.profile.plot` *Profile Fair Models with Respect to Tuning Parameters*

### Description

Visually explore various aspect of a model over the range of possible values of the tuning parameters that control its fairness.

### Usage

```
fairness.profile.plot(response, predictors, sensitive, epsilon,
  legend = FALSE, type = "coefficients", ...)
```

### Arguments

`response` a numeric vector, the response variable.

`predictors` a numeric matrix or a data frame containing numeric and factor columns; the predictors.

`sensitive` a numeric matrix or a data frame containing numeric and factor columns; the sensitive attributes.

`epsilon` a vector of positive numbers in [0, 1], the proportion of the explained variance that can be attributed to S. The default value is `seq(from = 0.00, to = 1, by = 0.02)`.

`legend` a logical value, whether to add a legend to the plot.

`type` a character string, either `"coefficients"` (the default) or `"variance"`.

`...` additional arguments passed to `nclm()`.

### Details

fairness.profile.plot() fits the model for all the values of the argument epsilon, and produces a profile plot of the regression coefficients or the proportion of explained variance.

If type = "coefficients", the coefficients of the model are plotted against the values of epsilon.

If type = "variance", the following quantities are plotted against the values of epsilon:

1. the proportion of variance explained by the sensitive attributes (with respect to the response;
2. the proportion of variance explained by the predictors (with respect to the response;
3. the proportion of variance explained by the sensitive attributes (with respect to the combined sensitive attributes and predictors).

### Value

A trellis object containing a **lattice** plot.

### Author(s)

Marco Scutari

### Examples

```
data(vur.test)
names(dev.cur())
fairness.profile.plot(response = vur.test$y, predictors = vur.test$X,
  sensitive = vur.test$S, type = "coefficients", legend = TRUE)
fairness.profile.plot(response = vur.test$y, predictors = vur.test$X,
  sensitive = vur.test$S, type = "variance", legend = TRUE)
```

## law.school.admissions *Law School Admission Council data*

### Description

Survey among students attending law school in the U.S. in 1991.

### Usage

```
data(law.school.admissions)
```

### Format

The data contains 20800 observations and the following variables:

- age, a continuous variable containing the student's age in years;
- decile1, a continuous variable containing the student's decile in the school given his grades in Year 1;

- decile3, a continuous variable containing the student's decile in the school given his grades in Year 3;
- fam_inc, a continuous variable containing student's family income bracket (from 1 to 5);
- lsat, a continuous variable containing the student's LSAT score;
- ugpa, a continuous variable containing the student's undergraduate GPA;
- gender, a factor with levels "female" and "male";
- race1, a factor with levels "asian", "black", "hisp", "other" and "white";
- cluster, a factor with levels "1", "2", "3", "4", "5" and "6" encoding the tiers of law school prestige;
- fulltime, a factor with levels "FALSE" and "TRUE", whether the student will work full-time or part-time;
- bar, a factor with levels "FALSE" and "TRUE", whether the student passed the bar exam on the first try.

## Note

The data set has been pre-processed as in Komiyama et al. (2018), with the following exceptions:

- DOB_yr, the year of birth, has been dropped because it is (nearly) perfectly collinear with age, and thus it is redundant;
- decile1b has been dropped because it is (nearly) perfectly collinear with decile1, and thus it is redundant.

In that paper, ugpa is the response variable, age and race1 are the sensitive attributes and the remaining variables are used as predictors.

## References

Sander RH (2004). "A Systemic Analysis of Affirmative Action in American Law Schools". Stanford Law Review, 57:367–483.

## Examples

```
data(law.school.admissions)

# short-hand variable names.
ll = law.school.admissions

m = nclm(response = ll[, "ugpa"],
      sensitive = ll[, c("age", "race1")],
      predictors = ll[, setdiff(names(ll), c("ugpa", "age", "race1"))],
      epsilon = 0.05)

summary(m)
```

## `methods for fair.model objects` *Extract information from fair.model objects*

### Description

Extract various quantities of interest from an object of class `fair.model`.

### Usage

```
# methods for all fair.model objects.
## S3 method for class 'fair.model'
coef(object, ...)
## S3 method for class 'fair.model'
residuals(object, ...)
## S3 method for class 'fair.model'
fitted(object, ...)
## S3 method for class 'fair.model'
sigma(object, ...)
## S3 method for class 'fair.model'
nobs(object, ...)
## S3 method for class 'fair.model'
print(x, digits, ...)
## S3 method for class 'fair.model'
summary(object, ...)
## S3 method for class 'fair.model'
all.equal(target, current, ...)

# methods for nclm objects.
## S3 method for class 'nclm'
predict(object, new.predictors, new.sensitive, ...)
## S3 method for class 'nclm'
summary(object, ...)
## S3 method for class 'nclm'
deviance(object, ...)
```

### Arguments

`object,x,target,current`
an object of class `fair.model` or `nclm`.

`digits` a non-negative integer, the number of significant digits.

`new.predictors` a numeric matrix or a data frame containing numeric and factor columns; the predictors for the new observations.

`new.sensitive` a numeric matrix or a data frame containing numeric and factor columns; the sensitive attributes for the new observations.

`...` additional arguments, currently ignored.

## national.longitudinal.survey *Income and Labour Market Activities*

### Description

Survey results from the U.S. Bureau of Labor Statistics to gather information on the labour market activities and other life events of several groups.

### Usage

```
data(national.longitudinal.survey)
```

### Format

The data contains 4908 observations and the following variables:

- `age`, a numeric variable containing the interviewee's age in years;
- `race`, a factor with 20 levels denoting various racial/ethnic origins;
- `gender`, a factor with levels `"Male"` and `"Female"`.
- `grade90`, a factor containing the highest completed school grade from "3RD GRADE" to "8TH YR COL OR MORE", with 18 levels;
- `income06`, a numeric variable, income in 2006 in 10000-USD units;
- `income96`, a numeric variable, income in 1996 in 10000-USD units;
- `income90`, a numeric variable, income in 1990 in 10000-USD units;
- `partner`, a factor encoding whether the interviewee has a partner, with levels `"No"` and `"Yes"`;
- `height`, a numeric variable, the height of the interviewee;
- `weight`, a numeric variable, the weight of the interviewee;
- `famsize`, a numeric variable, the number of family members;
- `genhealth`, a factor with levels `"Excellent"`, `"Very Good"`, `"Good"`, `"Fair"`, `"Poor"` encoding the general health status of the interviewee;
- `illegalact`, a numeric variable containing the number of illegal acts committed by the interviewee;
- `charged`, a numeric variable containing the number of illegal acts for which the interviewee has been charged;
- `jobsnum90`, a numeric value, the number of different jobs ever reported;
- `afqt89`, a numeric value, the percentile score of the "Profiles, Armed Forces Qualification Test" (AFQT);
- `typejob90`, a factor with 13 levels encoding different job types;
- `jobtrain90`, a factor with levels `"No"` and `"Yes"` encoding whether the job was classified as training.

### Note

The data set has been pre-processed differently from Komiyama et al. (2018). In particular:

- the variables `income96` and `income06` have been retained as alternative responses;
- the variables `height`, `weight`, `race`, `partner` and `famsize` have been retained;
- the variables `grade90` and `genhealth` are coded as ordered factors because they do not make sense on a numeric scale.

In that paper, `income90` is the response variable, `gender` and `age` are the sensitive attributes.

### References

U.S. Bureau of Labor Statistics: https://www.bls.gov/nls/

### Examples

```
data(national.longitudinal.survey)

# short-hand variable names.
nn = national.longitudinal.survey
# remove alternative response variables.
nn = nn[, setdiff(names(nn), c("income96", "income06"))]

m = nclm(response = nn[, "income90"],
      sensitive = nn[, c("gender", "age")],
      predictors = nn[, setdiff(names(nn), c("income90", "gender", "age"))],
      epsilon = 0.05)

summary(m)
```

---

## `nclm` *Nonconvex Optimization for Regression with Fairness Constraints*

---

### Description

Fair regression model based on nonconvex optimization from Komiyama et al. (2018).

### Usage

```
nclm(response, predictors, sensitive, epsilon)
```

### Arguments

| | |
|---|---|
| `response` | a numeric vector, the response variable. |
| `predictors` | a numeric matrix or a data frame containing numeric and factor columns; the predictors. |
| `sensitive` | a numeric matrix or a data frame containing numeric and factor columns; the sensitive attributes. |
| `epsilon` | a positive number in [0, 1], the proportion of the explained variance that can be attributed to `S`. |

### Details

The algorithm proposed by Komiyama et al. (2018) works like this:

1. regresses the predictors against the sensitive attributes;
2. constructs a new set of predictors that are decorrelated from the sensitive attributes using the residuals of this regression;
3. regresses the response against the decorrelated predictors and the sensitive attributes, while
4. bounding the proportion of variance the sensitive attributes can explain with respect to the overall explained variance of the model.

Both `sensitive` and `predictors` are standardized internally before estimating the regression coefficients, which are then rescaled back to match the original scales of the variables. `response` is only standardized if it has a variance smaller than `1`, as that seems to improve the stability of the solutions provided by the optimizer (as far as the data included in **fairml** are concerned).

### Value

`nclm()` returns an object of class `c("nclm","fair.model")`.

### Author(s)

Marco Scutari

### References

Komiyama J, Takeda A, Honda J, Shimao H (2018). "Nonconvex Optimization for Regression with Fairness Constraints". Proceedints of the 35th International Conference on Machine Learning (ICML), PMLR **80**:2737–2746.

## `vur.test` *Synthetic data set to test fair regression models*

### Description

This a synthetic data set used as a test case in the **fairml** package.

### Usage

```
data(vur.test)
```

### Format

The `vur.test` data set is a list with following three elements:

- `y`, the response variable;
- `X`, a numeric matrix containing 3 predictors called `X1`, `X2` and `X3`;
- `S`, a numeric matrix containing 3 sensitive attributes called `S1`, `S2` and `S3`.

## Note

This data set is called `vur.test` because it is generated from a *v*ery *u*nfair *r*egression model in which sensitive attributes explain the lion's share of the overall explained variance. The code used to generate the data is as follows.

```
library(mvtnorm)
sigma = matrix(0.3, nrow = 6, ncol = 6)
diag(sigma) = 1
n = 1000
X = rmvnorm(n, mean = rep(0, 6), sigma = sigma)
y = 2 + 2 * X[, 1] + 3 * X[, 2] + 4 * X[, 3] + 5 * X[, 4] +
        6 * X[, 5] + 7 * X[, 6] + rnorm(n, sd = 10)
S = X[, 4:6]
X = X[, 1:3]
colnames(X) = c(”X1”, ”X2”, ”X3”)
colnames(S) = c(”S1”, ”S2”, ”S3”)
vur.test = list(y = y, X = X, S = S)
```

## Examples

```
data(vur.test)
sensitive.attributes.model = lm(y ~ S, data = vur.test)
summary(sensitive.attributes.model)$r.squared
overall.model = lm(y ~ X + S, data = vur.test)
summary(overall.model)$r.squared
```

# Index