fairadapt: Causal Reasoning for Fair Data Pre-processing

Drago Plecko
ETH Zürich

Nicolas Bennett
ETH Zürich

Nicolai Meinshausen
ETH Zürich

Abstract

Machine learning algorithms are useful for various predictions tasks, but they can also learn how to discriminate, based on gender, race or some other sensitive attribute. This realization gave rise to the field of fair machine learning, which aims to measure and mitigate such algorithmic bias. This manuscript describes the implementation of the **fairadapt** R-package, a causal inference pre-processing method, which, using the causal graphical model, answers hypothetical questions of the form "What would my salary have been, had I been of a different gender/race?". Such counterfactual reasoning can help eliminate discrimination and help justify fair decisions.

Keywords: algorithmic fairness, causal inference, machine learning.

1. Introduction

Machine learning algorithms are now used for decision-making in socially sensitive situations, such as predicting credit-score ratings or recidivism during parole. Important early works noted that algorithms are capable of learning societal biases, for example with respect to race (Larson, Mattu, Kirchner, and Angwin 2016) or gender (Blau and Kahn 2003; Lambrecht and Tucker 2019). This realization started an important debate in the machine learning community about fairness of algorithms and their impact on decision-making.

The first step of fairness is defining and measuring discrimination. Some inuitive notions have been statistically formalized in order to provide fairness metrics. For example, the notion of *demographic parity* (Darlington 1971) requires the protected attribute A (gender/race/religion etc.) to be independent of a constructed classifier or regressor $\widehat{Y}$. Another notion, termed *equality of odds* (Hardt, Price, Srebro *et al.* 2016), requires the false positive and false negative rates of classifier $\widehat{Y}$ between different groups (females and males for example), written mathematically as $\widehat{Y} \perp\!\!\!\perp A \mid Y$. To this day, various different notions of fairness exist, which are sometimes incompatible (Corbett-Davies and Goel 2018), meaning not of all of them can be achieved for a predictor $\widehat{Y}$ simultaneously. There is no consensus on which notion of fairness is the correct one.

The discussion on algorithmic fairness is, however, not restricted to the machine learning domain. There are many legal and philosophical aspects that have arisen. For example, the legal distinction between disparate impact and disparate treatment (McGinley 2011) is important for assessing fairness from a judicial point of view. This in turn emphasizes the importance of the interpretation behind the decision-making process, which is often not the

case with black-box machine learning algorithms. For this reason, research in fairness through a causal inference lens has gained more attention.

There are several ways causal inference can help us understand and measure discrimination. The first is counterfactual reasoning (Galles and Pearl 1998), which allows us to argue what might have happened under different circumstances which did not occur. For example, we might ask whether a female candidate would had been employed, had she been male? This motivated another notion of fairness, termed *counterfactual fairness* (Kusner, Loftus, Russell, and Silva 2017), which states that the decision made should stay the same, even if we hypothetically changed someone's race or gender (written succintly as $\widehat{Y}(a) = \widehat{Y}(a')$ in the potential outcome notation). Further, important work has been done in order to decompose the parity gap measure (used for assesing demographic parity), $\mathbb{P}(\widehat{Y} = 1 \mid A = a) - \mathbb{P}(\widehat{Y} = 1 \mid A = a')$, into the direct, indirect and spurious components. Lastly, the work of Kilbertus, Carulla, Parascandolo, Hardt, Janzing, and Schölkopf (2017) introduces the so-called resolving variables, in order to relax the possibly prohibitively strong notion of demographic parity. This manuscript describes the details of the fair data adaptation method (Plecko and Meinshausen 2020). The approach aims to combine the notions of counterfactual fairness and resolving variables and to explicitly compute counterfactul values of individuals. The implementation is available on CRAN as the **fairadapt** package.

We note that as of the day of writing of the manuscript, there are only 4 CRAN packages related fair machine learning. The **fairml** package implements the non-convex method of Komiyama, Takeda, Honda, and Shimao (2018). Packages **fairness** and **fairmodels** serve as diagnostic tools for measuring algorithmic bias, together with an implementation of several pre and post-processing methods for bias mitigation. However, the **fairadapt** package is the only causal method. Even though many papers on the topic have been published, the fairness domain is still lacking good quality implementations of the existing methods.

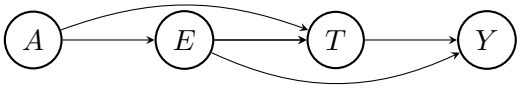
The rest of the manuscript is organized as follows. In Section 2 we describe the methodology behind **fairadapt**, together with quickly reviewing some of the important concepts of causal inference. In Section 3 we discuss the implementation details and guide the user as to how to use the package. In Section 4 we illustrate the usage of **fairadapt** by using a large, real-world dataset for a hypothetical fairness application. In Section 5 we explain some important extensions, such as Semi-Markovian models and resolving variables.

2. Methodology

We start by describing the basic idea of **fairadapt** in a nutshell, followed by the precise mathematical formulation.

2.1. Example: university admission

Consider the following example. Variable A is the protected attribute, in this case gender ($A = a$ corresponding to females, $A = a'$ to males). Let E be educational achievement (measured for example by grades achieved in school) and T the result of an admissions test for further education. Let Y be the outcome of interest (final score) upon which admission to further education is decided. Edges in the graph indicate how variables affect each other.



Attribute A, gender, has a causal effect on variables E, T and Y, and we wish to eliminate this effect. For each individual with observed values (a, e, t, y) we want to find a mapping

$$(a, e, t, y) \longrightarrow (a^{(fp)}, e^{(fp)}, t^{(fp)}, y^{(fp)}),$$

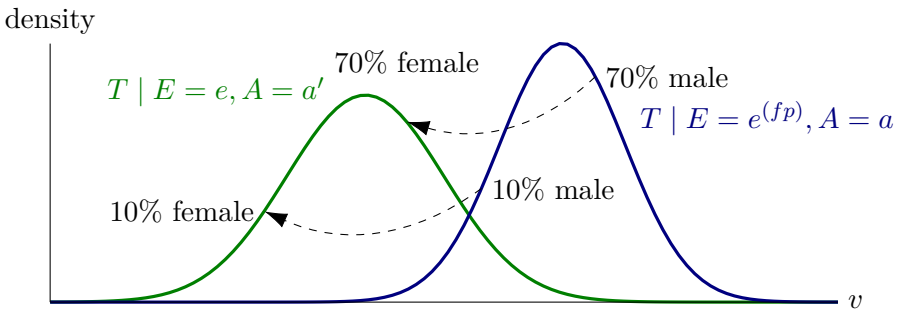
which finds the value the person would have obtained in a world where everyone is female. Explicitly, for a male person with education value e, we give it the transformed value $e^{(fp)}$ chosen such that

$$\mathbb{P}(E \geq e \mid A = a') = \mathbb{P}(E \geq e^{(fp)} \mid A = a).$$

The main idea is that the *relative educational achievement within the subgroup* would stay the same if we changed someone's gender. If you are male and you have a higher educational achievement than 60% of all males in the dataset, we assume you would be better than 60% of females had you been female[1]. After computing everyone's education (in the 'female' world), we continue by computing the transformed test score values $T^{(fp)}$. The approach is again similar, but this time we condition on educational achievement. That is, a male with values $(E, T) = (e, t)$ is assigned a test score $t^{(fp)}$ such that

$$\mathbb{P}(T \geq t \mid E = e, A = a') = \mathbb{P}(T \geq t^{(fp)} \mid E = e^{(fp)}, A = a),$$

where the value $e^{(fp)}$ was obtained in the previous step. The step can be visualized as follows[2]



In the last step, the outcome variable Y needs to be adjusted. The adaptation is based on the values of education and the test score. The transformed value $y^{(fp)}$ of $Y = y$ would satisfy

$$\mathbb{P}(Y \geq y \mid E = e, T = t, A = a') = \mathbb{P}(Y \geq y^{(fp)} \mid E = e^{(fp)}, T = t^{(fp)}, A = a). \quad (1)$$

This way of counterfactual correction is known as *recursive substitution* (Pearl 2009, Chapter 7).

[1]This assumption is empirically untestable, since it is impossible to observe both a female and a male version of the same individual.

[2]In the visualization, the test scores of male applicants have higher values. We emphasize this is in no way a view implied by the authors, simply a currently observed societal bias in certain university admission datasets.

We next describe the approach from above formally. The reader who is not interested in the mathematical detail is encouraged to go straight to Section 3. We start by introducing an important causal inference concept, related to our discussion, namely the *structural causal model*. A structural causal model (SCM) is a 4-tuple $< V, U, \mathcal{F}, P(u) >$, where

- $V = \{V_1, ..., V_n\}$ is the set of observed (endogeneous) variables
- $U = \{U_1, ..., U_n\}$ are latent (exogeneous) variables
- $\mathcal{F} = \{f_1, ..., f_n\}$ is the set of functions determining V, $v_i \leftarrow f_i(\text{pa}(v_i), u_i)$, where $\text{pa}(V_i) \subset V, U_i \subset U$ are the functional arguments of f_i
- $P(u)$ is a distribution over the exogeneous variables U.

We note that any particular SCM is accompanied by a graphical model $\mathcal{G}$ (a directed acyclic graph), which summarizes which functional arguments are necessary for computing the values of each V_i (that it is, how variables affect each other). We assume throughout, without loss of generality, that

(i) $f_i(\text{pa}(v_i), u_i)$ is increasing in u_i for every fixed $\text{pa}(v_i)$

(ii) exogeneous variables U_i are uniformly distributed on $[0, 1]$

We first discuss the so-called Markovian case in which all exogeneous variables U_i are mutually independent. Some relevant extensions, like the Semi-Markovian case (where U_i variables are allowed to have mutual dependencies) and the case of so called *resolving variables*, are discussed in Section 5.

2.2. Basic formulation - Markovian SCMs

Suppose that Y taking values in $\mathbb{R}$ is an outcome of interest and A the protected attribute taking two values a, a'. Our goal is to describe a pre-processing method which transform the entire data V into its fair version $V^{(fp)}$. This is done by computing the counterfactual values $V(A = a)$ which would have been obtained by the individuals, had everyone had the same protected attribute $A = a$.

More precisely, going back to the *university admission* example above, we want to "equate" the distributions

$$V_i \mid \text{pa}(V_i), A = a \text{ and } V_i \mid \text{pa}(V_i), A = a'. \tag{2}$$

In words, we want the distribution of V_i to be the same for the female and male applicants, for every variable V_i. Since each function f_i of the original SCM is reparametrized so that $f_i(\text{pa}(v_i), u_i)$ is increasing in u_i for every fixed $\text{pa}(v_i)$, and also that U_i variables are uniformly distributed on $[0, 1]$. Then the U_i variables can be seen as the latent *quantiles*. Our algorithm proceedes as follows:

The f_i assignment functions of the SCM are of course unknown, but are learned non-parametrically at each step. Notice that Algorithm 1 is computing the counterfactual values $V(A = a)$ under the $do(A = a)$ intervention for each individual, while keeping the latent quantiles U fixed. In the case of continuous variables, the latent quantiles U can be determined exactly, while for

Input: V, causal graph $\mathcal{G}$
set $A \leftarrow a$ for everyone
for $V_i \in \text{de}(A)$ *in topological order* **do**
  learn the assignment function $V_i \leftarrow f_i(\text{pa}(V_i), U_i)$
  infer the quantiles U_i associated with the variable V_i
  transform the values of V_i by using the quantile and the transformed parents (obtained in previous steps) $V_i^{(fp)} \leftarrow f_i(\text{pa}(V_i)^{(fp)}, U_i)$
end
return $V^{(fp)}$

Algorithm 1: Fair Data Adaptation

the discrete case, this is more subtle and described in detail in the original fair data adaptation manuscript (Plecko and Meinshausen 2020, Section 5).

3. Implementation

The implementation is based on the main function `fairadapt()`, which returns an S3 object of class `"fairadapt"`. We list the most important arguments of the function and then show how these should be specified:

- `formula`, argument of type `formula` specifies the dependent and explanatory variables
- `adj.mat` argument of type `matrix` encodes the adjacency matrix
- `train.data`, `test.data` of type `data.frame`
- `prot.attr` of type `character` is of length one and names the protected attribute.

```
R> uni.adj.mat <- array(0, dim = c(4, 4))
R> colnames(uni.adj.mat) <- rownames(uni.adj.mat) <-
+   c("gender", "edu", "test", "score")
R>
R> uni.adj.mat["gender", c("edu", "test")] <-
+   uni.adj.mat["edu", c("test", "score")] <-
+   uni.adj.mat["test", "score"] <- 1L
R>
R> nsamp <- 200
R>
R> FA.basic <- fairadapt(score ~ .,
+   train.data = uni_admission[1:nsamp, ],
+   test.data = uni_admission[(nsamp+1):(2*nsamp), ],
+   adj.mat = uni.adj.mat, prot.attr = "gender", res.vars = NULL,
+   visualize.graph = F, quant.method = fairadapt:::rangerQuants)
R>
R> FA.basic

Fairadapt result

Call:
```

```
 score ~ .

Protected attribute:                  gender
Protected attribute levels:           0, 1
Number of training samples:           200
Number of test samples:               200
Number of independent variables:      3
Total variation (before adaptation):  -0.6757414
Total variation (after adaptation):   -0.07889245
```

The `"fairadapt"` `S3` class has several associated generics and methods. For instance, `print(FA.basic)` shows some information about the object call, such as number of variables, the protected attribute and also the total variation before and after adaptation, defined as (Y denoting the outcome variable)

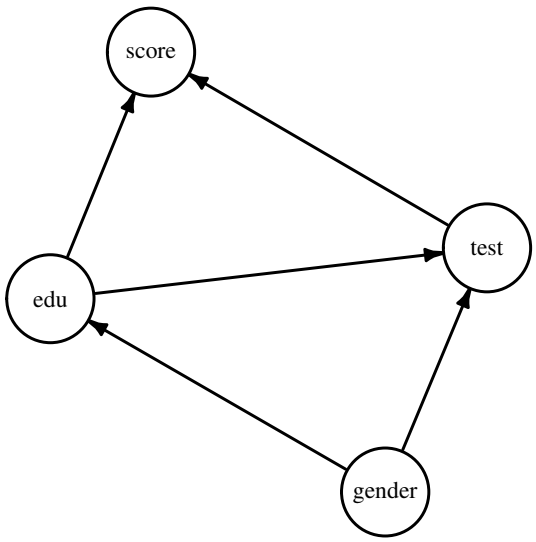
$$\mathbb{E}[Y \mid A = a] - \mathbb{E}[Y \mid A = a'] \text{ and } \mathbb{E}[Y^{(fp)} \mid A = a] - \mathbb{E}[Y^{(fp)} \mid A = a'],$$

respectively. The total variation, in the case of binary Y, corresponds to the parity gap.

3.1. Specifying the graphical model

The **fairadapt** supposes the underlying graphical model $\mathcal{G}$ is known. The model is specified by the adjacency matrix. For example, suppose we take the causal graph $\mathcal{G}$ of the university admission example above. For such a graph, we construct the adjacency matrix and the graph with the `GraphModel()` convenience function that builds on top of the **igraph** package.

```
R> toy.graph <- graphModel(uni.adj.mat)
R> plot(toy.graph, vertex.size = 40, vertex.label.cex = 0.5,
+   vertex.label.color = "black")
```



3.2. Quantile learning step

	Random Forests	Neural Networks	Linear Regression
R-package	**ranger**	**mcqrnn**	**quantreg**
`quant.method`	`rangerQuants`	`mcqrnnQuants`	`linearQuants`
complexity	$O(np \log n)$	$O(npn_{\text{epochs}})$	$O(p^2 n)$
default parameters	$ntrees = 500$ $mtry = \sqrt{p}$	2-layer fully connected feed-forward network	`"br"` method of Barrodale and Roberts used for fitting
$T(200, 4)$	0.4 sec	89 sec	0.3 sec
$T(500, 4)$	0.9 sec	202 sec	0.5 sec

Table 1: Summary table of different quantile regression methods. n is the number of samples, p number of covariates, n_{epochs} number of training epochs for the NN. $T(n, 4)$ denotes the runtime of different methods on the university admission dataset, with n training and testing samples.

We describe the training step using the `fairadapt()` function. The `fairadapt()` function can be used in two slightly distinct ways. The first option is by specifying training and testing data at the same time. The data adaptation is then applied to the combination of train and test data, in order to learn the latent quantiles as precisely as possible (with the exception of label Y which is unavailable on the test set). The second option is use only the `train.data` argument when calling `fairadapt()`, after which the `predict()` function can be used to adapt test data at a later stage.

We note that `train.data` and `test.data` need to have column names which appear in the names of the adjacency matrix `colnames(adj.mat)`. The protected attribute A is given as a character vector `prot.attr` of length one.

The quantile learning step in Algorithm 1 can be done using three different methods:

- Quantile Regression Forests (Meinshausen 2006)
- Non-crossing quantile neural networks (Cannon 2018)
- Linear Quantile Regression (Koenker and Hallock 2001)

The summary of the various differences between the methods is given in Table 1.

The choice of quantile learning method is done by specifying the `quant.method` argument, which is of class `function` and constructs the quantile regression object. For more details and an example, see `?rangerQuants`. Together with the `quant.method` constructor, `S3`-dispatch is used for inferring the quantiles. This allows the user to specify their own quantile learning methods easily.

We quickly discuss the quantile learning methods included in the package. Using the linear quantile method is the fastest option. However, it cannot handle the non-parametric case. For a non-parametric approach and mixed data, the RF approach is well-suited. The neural network approach is, comparatively, substantially slower than the forest/linear case and does not scale well to large sample sizes. Generally, we recommend using the forest based approach,

because of the non-parametric nature and computational speed. However, we note that for smaller sample sizes, the neural network approach might in fact be the best option.

3.3. Fair-twin inspection

The university admission example presented in Section 2 demonstrates how we can compute counterfactual values for an individual while preserving their relative educational achievement. In particular, for a male student with values (a, e, t, y), we compute his "fair-twin" values $(a^{(fp)}, e^{(fp)}, t^{(fp)}, y^{(fp)})$ - the values the student would have obtained, had he been female. To explicitly compare a person to their hypothetical fair-twin, we use the `fairTwins()` function, applied to an object of class `"fairadapt"`:

```
R> fairTwins(FA.basic, train.id = seq.int(1L, 5L, 1L))
```

```
  gender      score score_adapted        edu edu_adapted         test
1      1  1.9501728    1.34359000  1.3499572  0.69580978  1.617739679
2      0 -2.3502495   -2.35024955 -1.9779234 -1.97792341 -3.121796235
3      1  0.6285619   -0.04234933  0.6263626 -0.29744386  0.530034686
4      1  0.7064857    0.13173095  0.8142112 -0.02637841  0.004573003
5      1  0.3678313    0.23076887  1.8415242  0.90893547  1.193677123
  test_adapted
1    0.5555573
2   -3.1217962
3   -0.5822567
4   -0.7973851
5    0.3267990
```

In this example, we compute the values in a "female" world. Therefore, for female applicants, the values stay the same, while for male applicants the values are adapted, as can be seen from the output.

4. Illustration

Here we describe an example of a possible real-world use of **fairadapt**. Suppose that after a legislative change the US government has decided to adjust the salary of all of its female employees in order to remove both disparate treatment and disparate impact effects. To this end, the government wants to compute the counterfactual salary values of all female employees, that is the salaries that female employees would obtain, had they been male.

To do this, the government is using the from the 2018 American Community Survey by the US Census Bureau. We load the pre-processed version of the dataset:

```
R> dat <- gov_census
R> print(head(dat))
```

```
      sex age  race hispanic_origin citizenship nativity  marital family_size
1:   male  64 black              no           1   native  married           2
```

```
2: female  54 white              no           1   native  married           3
3:   male  38 black              no           1   native  married           3
4: female  41 asian              no           1   native  married           3
5: female  40 white              no           1   native  married           4
6: female  46 white              no           1   native divorced           3
   children education_level english_level salary hours_worked weeks_worked
1:        0              20             0  43000           56           49
2:        1              20             0  45000           42           49
3:        1              24             0  99000           50           49
4:        1              24             0  63000           50           49
5:        2              21             0  45200           40           49
6:        1              18             0  28000           40           49
   occupation industry economic_region
1:    13-1081     928P       Southeast
2:    29-2061     6231       Southeast
3:    25-1000    611M1       Southeast
4:    25-1000    611M1       Southeast
5:    27-1010    611M1       Southeast
6:    43-6014     6111       Southeast
```

```
R> # group the columns
R> prot.attr <- "sex"
R> dmgraph <- c("age", "race", "hispanic_origin", "citizenship", "nativity",
+   "economic_region")
R> fam <- c("marital", "family_size", "children")
R> edu <- c("education_level", "english_level")
R> work <- c("hours_worked", "weeks_worked", "occupation", "industry")
R> out <- "salary"
```

The hypothesized causal graph for the dataset is given in Figure 1. We construct the causal graph and the confounding matrix:

```
R> col.names <- c(prot.attr, dmgraph, fam, edu, work, out)
R>
R> adj.mat <- cfd.mat <- array(0, dim = c(length(col.names), length(col.names)))
R> colnames(adj.mat) <- rownames(adj.mat) <-
+   colnames(cfd.mat) <- rownames(cfd.mat) <-
+   col.names
R>
R> adj.mat[prot.attr, c(fam, edu, work, out)] <-
+   adj.mat[dmgraph, c(fam, edu, work, out)] <-
+   adj.mat[fam, c(edu, work, out)] <-
+   adj.mat[edu, c(work, out)] <-
+   adj.mat[work, out] <-
+   cfd.mat[prot.attr, dmgraph] <- cfd.mat[dmgraph, prot.attr] <- 1L
R>
R> census.graph <- graphModel(adj.mat, cfd.mat)
```

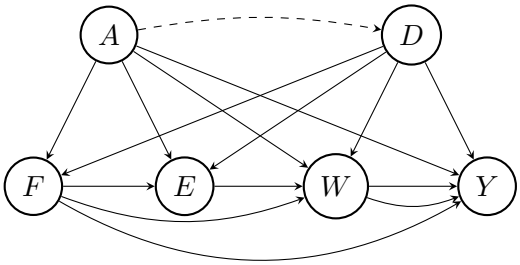
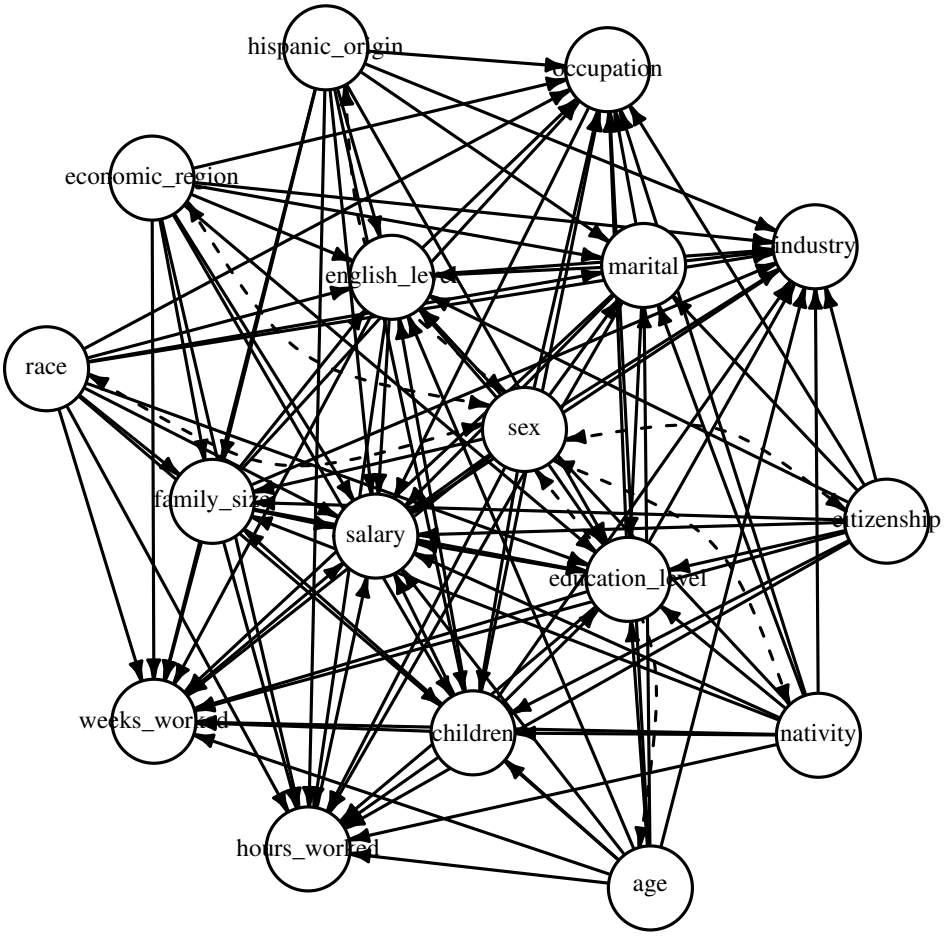


Figure 1: The causal graph for the Government-Census dataset. D are demographic features, A is gender, F is marital and family information, E education, W work-related information, Y the salary, which is also the outcome of interest.

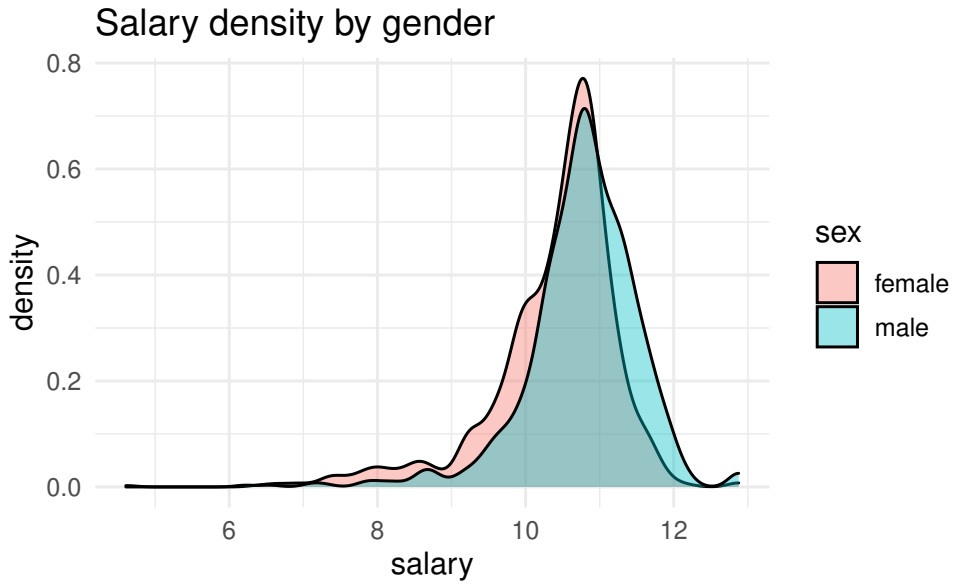
```
R> plot(census.graph, vertex.size = 20, vertex.label.cex = 0.5,
+   vertex.label.color = "black")
```



Before applying `fairadapt()`, we first log-transform the salaries and look at the densities by gender group

```
R> # log-transform the salaries
R> dat$salary <- log(dat$salary)
```

```
R>
R> # plot density before adaptation
R> nsamples <- 2000
R>
R> ggplot(dat[1:nsamples], aes(x = salary, fill = sex)) +
+   geom_density(alpha = 0.4)  + theme_minimal() +
+   ggtitle("Salary density by gender")
```



There is a clear shift between the two genders, meaning that `male` employees are currently treated better than `female` employees. However, there could be differences in `salary` which are not due to gender inequality, but have to do with the economic region in which the employee works. This needs to be accounted for as well, i.e. the difference between economic regions is not to be removed. To solve the problem, the US governemnt applies **fairadapt**:

```
R> FA.govcensus <- fairadapt(salary ~ ., train.data = dat[1:nsamples],
+                            adj.mat = adj.mat, prot.attr = prot.attr,
+                            visualize.graph = F)
```

After applying the adaptation, we inspect whether the problem has improved. The densities after adaptation can be plotted using the `autoplot()` function:

```
R> autoplot(FA.govcensus, when = "after") +
+   ggtitle("Adapted salary density by gender")
```

If we obtain additional testing data, and wish to adapt it as well, we can use the `predict()` function:

```
R> new.test <- dat[seq.int(nsamples + 1L, nsamples + 10L, 1L)]
R> adapt.test <- predict(FA.govcensus, newdata = new.test)
R> head(adapt.test)
```

```
      sex age  race hispanic_origin citizenship nativity marital family_size
1: female  52 white              no           1   native married           3
2: female  31 white              no           1   native married           5
3: female  53 white              no           1   native married           2
4: female  53 black              no           1   native married           2
5: female  23 white              no           1   native married           2
6: female  49 white             yes           1   native married           7
   children education_level english_level   salary hours_worked weeks_worked
1:        1              21             0 11.91839           40           49
2:        3              22             0 10.77896           40           49
3:        0              22             0 11.40756           40           49
4:        0              21             0 11.28978           40           49
5:        0              22             0 10.46310           15           49
6:        4              22             0 10.69194           40           49
   occupation industry economic_region
1:    13-1082     92M2       Southeast
2:    25-2020     6111       Southeast
3:    25-2050    51912       Southeast
4:    11-91XX     928P       Southeast
5:    43-9XXX 928110P2       Southeast
6:    25-2020     6111       Southeast
```

Finally, we can do fair-twin inspection using the `fairTwins()` function, to see how feature values of individual employees have changed:

```
R> inspect.cols <- c("sex", "age", "education_level", "salary")
R> fairTwins(FA.govcensus, train.id = 1:5, cols = inspect.cols)
```

```
     sex age age_adapted education_level education_level_adapted   salary
1   male  64          64              20                       20 10.66896
2 female  54          54              20                       20 10.71442
3   male  38          38              24                       24 11.50288
4 female  41          41              24                       24 11.05089
5 female  40          40              21                       21 10.71885
  salary_adapted
1       10.49035
2       10.71442
3       11.60824
4       11.05089
5       10.71885
```

The values are unchanged for the female individuals. Note that `age` does not change for any individual, since it is not a descendant of A. However, variables `education_level` and `salary` do change for males, as they are descendants of A.

The variable `hours_worked` is also a descendant of A. However, one might argue that this variable should not be adapted in the procedure, that is, that it should remain the same, even if we hypothetically change the person's gender. This is the idea behind *resolving variables*, introduced in the next section.

5. Extensions

5.1. Adding resolving variables

Kilbertus *et al.* (2017) discuss that in some situations the protected attribute A can affect variables in a non-discriminatory way. For instance, in the Berkeley admissions dataset (Bickel, Hammel, and O'Connell 1975) we observe that females often apply for departments with lower admission rates and consequently have a lower admission probability. However, we perhaps would not wish to account for this difference in the adaptation procedure if we were to argue that department choice is a choice everybody is free to make. This motivated the following reasoning, found in Kilbertus *et al.* (2017). A variable R is called resolving if

(i) $R \in \text{de}(A)$, where $\text{de}(A)$ are the descendants of A in the causal graph $\mathcal{G}$

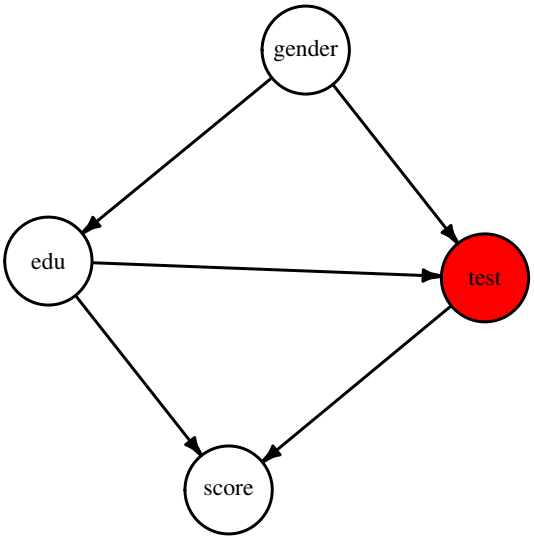
(ii) the causal effect of A on R is considered to be non-discriminatory

In presence of resolving variables, we compute the counterfactual under a more complicated intervention $\text{do}(A = a, R = R(a'))$. The potential outcome value $V(A = a, R = R(a'))$ is obtained by setting $A = a$ and computing the counterfactual while keeping the values of resolving variables to those they *attained naturally*. This is a nested counterfactual and the difference in Algorithm 1 is simply that resolving variables R are skipped over in the for-loop. We run the following code to compute the fair adaptation with the variable `test` being resolving in the `uni_admission` dataset

```
R> FA.resolving <- fairadapt(score ~ .,
+   train.data = uni_admission[1:nsamp, ],
+   test.data = uni_admission[(nsamp+1):(2*nsamp), ],
+   adj.mat = uni.adj.mat, prot.attr = "gender", res.vars = "test",
+   visualize.graph = F)
R>
R> FA.resolving

Fairadapt result

Call:
 score ~ .

Protected attribute:                  gender
Protected attribute levels:           0, 1
Resolving variables:                  test
Number of training samples:           200
Number of test samples:               200
Number of independent variables:      3
Total variation (before adaptation):  -0.6757414
Total variation (after adaptation):   -0.3893239
```

We note that the total variation in this case is larger than in the `FA.basic` example from Section 3, with no resolvers. The intuitive reasoning here is that resolving variables allow for some discrimination, so we expect to see a larger total variation between the groups. Finally, we can visualize the graph

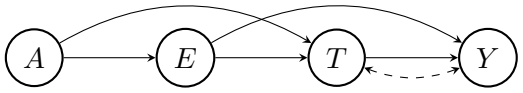
```
R> plot(graphModel(uni.adj.mat, res.vars = "test"),
+   vertex.size = 40, vertex.label.cex = 0.5,
+   vertex.label.color = "black")
```



which shows a different color for the resolving variable `test`. The resolving variables are red-colored in order to be distinguished from other variables.

5.2. Semi-Markovian and topological ordering variant

In Section 2 we were concerned with the Markovian case, which assumes that all exogeneous variables U_i are mutually independent. However, in practice this need not be the case. If there are mutual dependencies between the U_is, we are dealing with a so-called Semi-Markovian model. These dependencies between latent variables are represented by dashed, bidirected arrows in the causal diagram. In the university admission example, suppose we had that $U_{\text{test}} \not\perp\!\!\!\perp U_{\text{score}}$, meaning that latent variables corresponding to variable test and final score are correlated. Then the graphical model would be represented as



There is an important difference in the adaptation procedure for Semi-Markovian case: when inferring the latent quantiles U_i of variable V_i, in the Markovian case, only the direct parents $\text{pa}(V_i)$ are needed. In the Semi-Markovian case, due to correlation of latent variables, using only the $\text{pa}(V_i)$ can lead to biased estimates of the U_i. Instead, the set of direct parents needs to be extended, described in detail in (Tian and Pearl 2002). We briefly sketch the argument. Let the *C-components* be a partition of the set V, such that each $C-component$ contains a set of variables which are mutually connect by bidirected arrows. Let $C(V_i)$ denote the whole C-component of variable V_i. We then define the set of extended parents

$$\text{Pa}(V_i) := (C(V_i) \cup pa(C(V_i))) \cap \text{an}(V_i),$$

where $\text{an}(V_i)$ are the ancestors of V_i. The adaptation procedure in the Semi-Markovian case remains the same as in Algorithm 1, with the difference that the set of direct parents $\text{pa}(V_i)$ is replaced by $\text{Pa}(V_i)$ at each step.

To include the bidirected confounding arcs in the adaptation, we use the `cfd.mat` argument of type `matrix` such that

- `cfd.mat` has the same dimension, column and row names as `adj.mat`
- `cfd.mat` is symmetric and setting `cfd.mat["Vi", "Vj"] <- cfd.mat["Vj", "Vi"] <- 1L` indicates that there is a bidirected edge between variables V_i and V_j.

Alternatively, instead of using the extended parent set $\text{Pa}(V_i)$, we can use the "largest possible" set of parents, namely the ancestors $\text{an}(V_i)$. This approach is implemented, and the user only needs to specify the topological ordering. This is done by specifying the `top.ord` argument which is a `character` vector, containing the correct ordering of the names appearing in `names(train.data)`.

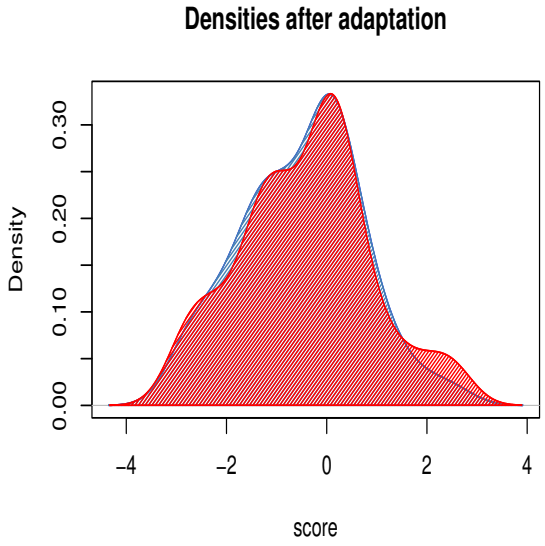
The following code runs the adaptation in the Semi-Markovian case:

```
R> uni.cfd.mat <- array(0, dim = c(4, 4))
R> colnames(uni.cfd.mat) <- rownames(uni.cfd.mat) <- colnames(uni.adj.mat)
```

```
R>
R> uni.cfd.mat["test", "score"] <- uni.cfd.mat["score", "test"] <- 1L
R> FA.semimarkov <- fairadapt(score ~ .,
+   train.data = uni_admission[1:nsamp, ],
+   test.data = uni_admission[(nsamp+1):(2*nsamp), ],
+   adj.mat = uni.adj.mat, cfd.mat = uni.cfd.mat, prot.attr = "gender",
+   visualize.graph = F)
```

We visualize the graph that was used for the adaptation.

```
R> plot(FA.semimarkov, graph = T, vertex.size = 40,
+   vertex.label.cex = 0.5, vertex.label.color = "black")
```



5.3. Questions of identifiability

So far, we have not discussed whether it is always possible to do the counterfactual inference described in the paper. In the causal literature, an intervention is *identifiable* if it can be computed uniquely using the data and the assumptions encoded in the graphical model $\mathcal{G}$. The important result by Tian and Pearl (2002) states that an intervention $\text{do}(X = x)$ on a singleton variable X is identifiable if and only if there is no bidirected path between X and $\text{ch}(X)$. Therefore, the intervention is identifiable if

- the model is Markovian
- the model is Semi-Markovian and
  - there is no bidirected path between A and $\text{ch}(A)$, and
  - there is no bidirected path between R_i and $\text{ch}(R_i)$ for any resolving variable R_i.

Based on this, the `fairadapt()` function sometimes returns a error, if the specified intervention is not possible to compute. One additional limitation is that **fairadapt** currently does

not support *front-door identification* (Pearl 2009, Chapter 3), but we hope to include this in a future version.

References

Bickel PJ, Hammel EA, O'Connell JW (1975). "Sex bias in graduate admissions: Data from Berkeley." *Science*, **187**(4175), 398–404.

Blau FD, Kahn LM (2003). "Understanding international differences in the gender pay gap." *Journal of Labor economics*, **21**(1), 106–144.

Cannon AJ (2018). "Non-crossing nonlinear regression quantiles by monotone composite quantile regression neural network, with application to rainfall extremes." *Stochastic environmental research and risk assessment*, **32**(11), 3207–3225.

Corbett-Davies S, Goel S (2018). "The measure and mismeasure of fairness: A critical review of fair machine learning." *arXiv preprint arXiv:1808.00023*.

Darlington RB (1971). "ANOTHER LOOK AT "CULTURAL FAIRNESS" 1." *Journal of Educational Measurement*, **8**(2), 71–82.

Galles D, Pearl J (1998). "An axiomatic characterization of causal counterfactuals." *Foundations of Science*, **3**(1), 151–182.

Hardt M, Price E, Srebro N, *et al.* (2016). "Equality of opportunity in supervised learning." In *Advances in neural information processing systems*, pp. 3315–3323.

Kilbertus N, Carulla MR, Parascandolo G, Hardt M, Janzing D, Schölkopf B (2017). "Avoiding discrimination through causal reasoning." In *Advances in Neural Information Processing Systems*, pp. 656–666.

Koenker R, Hallock KF (2001). "Quantile regression." *Journal of economic perspectives*, **15**(4), 143–156.

Komiyama J, Takeda A, Honda J, Shimao H (2018). "Nonconvex optimization for regression with fairness constraints." In *International conference on machine learning*, pp. 2737–2746. PMLR.

Kusner MJ, Loftus J, Russell C, Silva R (2017). "Counterfactual fairness." In *Advances in Neural Information Processing Systems*, pp. 4066–4076.

Lambrecht A, Tucker C (2019). "Algorithmic bias? an empirical study of apparent gender-based discrimination in the display of stem career ads." *Management Science*, **65**(7), 2966–2981.

Larson J, Mattu S, Kirchner L, Angwin J (2016). "How we analyzed the COMPAS recidivism algorithm." *ProPublica (5 2016)*, **9**.

McGinley AC (2011). "Ricci v. DeStefano: Diluting Disparate Impact and Redefining Disparate Treatment." *Nev. LJ*, **12**, 626.

Meinshausen N (2006). "Quantile regression forests." *Journal of Machine Learning Research*, **7**(Jun), 983–999.

Pearl J (2009). *Causality*. Cambridge University press.

Plecko D, Meinshausen N (2020). "Fair Data Adaptation with Quantile Preservation." *Journal of Machine Learning Research*, **21**, 1–44.

Tian J, Pearl J (2002). "A general identification condition for causal effects." In *Aaai/iaai*, pp. 567–573.

Affiliation:

Drago Plecko
ETH Zürich
Seminar for Statistics Rämistrasse 101 CH-8092 Zurich
E-mail: drago.plecko@stat.math.ethz.ch

Nicolas Bennett
ETH Zürich
Seminar for Statistics Rämistrasse 101 CH-8092 Zurich
E-mail: nicolas.bennett@stat.math.ethz.ch

Nicolai Meinshausen
ETH Zürich
Seminar for Statistics Rämistrasse 101 CH-8092 Zurich
E-mail: meinshausen@stat.math.ethz.ch