Fairness in Machine Learning

Cornelius Braun January 3rd 2024

In a previous blog post, we explained the plenitude of human biases that are often present in real-world data sets.

Since practitioners may be forced to work with biased data, it is crucial to know about ways in which the fairness of model decisions can nevertheless be guaranteed. Thus, in this post, I explain the most important ideas around fairness in machine learning (ML). This includes a short summary of the main metrics to measure the fairness of your model decisions and an overview of tools that can help you guarantee or improve your model's fairness.

1. Overview

By 2023, ML models are running a lot of processes behind the scenes in most of our daily lives. If you are reading this, there is a good chance that you have been in contact w outputs of an ML model today, for instance, when you unlocked your phone using fa recognition, or when you played around with a large language model such as GPT.

Menu

While the above examples may not be critical applications, ML models are likely used in sensitive areas of your life as well. For instance, ML models are now used to help practitioners to determine treatment strategies for patients, to propose sentence lengths for criminals, or to select job applicants. While mathematical models ideally make the decisions in each of these examples more objective, this is unfortunately not always the case in practice. Famous examples of biased and unfair ML models include:

1. The COMPAS algorithm predicted higher recidivism rates for African-American criminals than Caucasians with similar criminal records (Angwin et al., 2016).
2. The job application platform Xing ranked male applicants with similar qualifications higher than female ones (Lahoti et al., 2019).
3. Google Translate translated English, gender-neutral, sentences to Turkish phrases that carried heavy gender stereotypes, such as "the doctor" being translated to the male form of the noun (Caliskan et al., 2017), as illustrated in Fig. 1.

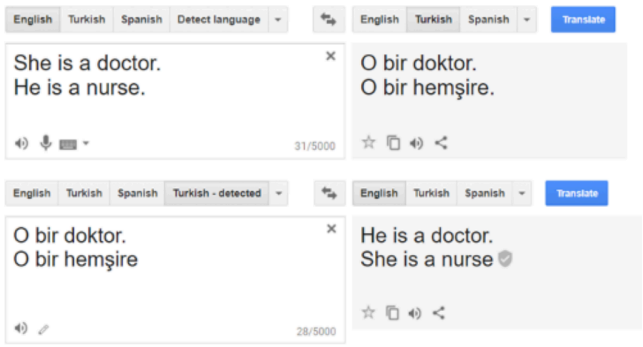

Figure 1: **Google Translate making gender-biased translations. Taken from Barocas et al. (2017).** *Note:* **The issue has been fixed since the publication of Caliskan et al. (2017), so you will not be able to reproduce this behavior.**

The reason behind such problematic model predictions is typically the **bias of models or data** (Mehrabi et al., 2021). There are many types of biases and even more reasons why they can occur in mathematical model predictions. Some of the most relevant biases in ML include dataset bias, or bias due to badly-chosen objective functions that weigh prediction errors more for one group than for another (Suresh & Guttag, 2021). If you want to know more about biases in ML, I recommend an earlier article on our blog that talks about this in more detail.

By 2023, the ML community has recognized the problem and there is an increasing interest in research on how fair model predictions can be guaranteed. For instance, most novel papers from major research companies such as Meta or Google feature mandatory model cards that provide information on fairness-related aspects such as training data distribution. To give you a better understanding of the research of this important subject (and tools that help you in development), I provide a short overview of the current state-of-the-art in this post. The structure of this post is as follows:

1. Introduction
2. What is fairness
3. Common pitfalls
4. Methods to improve fairness in ML

2. What is fairness?

Before we look at ways in which ML models can be made fairer, I will try to give you an understanding of what fairness means. Fairness is a concept that you encounter in many areas, from philosophy to law and mathematics, and each of these areas has come up with (several of) their own definitions of the matter. Yet, all of these definitions share the same fundamental idea: *a fair process shall treat all involved parties justly, and equally*.

You may wonder how this simple idea can be transferred to the world of ML, with its black-box models and huge datasets. And you are right, it is not easy. There exist plenty of definitions/criteria for fairness in ML, which all come with advantages and disadvantages. What most approaches have in common, is that they focus on **quantitative fairness criteria** (Caton & Haas, 2020). The benefit of using quantitative measures is that they permit to directly quantify the performance of a model concerning that metric. Thus, different interventions can be compared more objectively, and model fairness can be improved by treating it as part of the optimization problem.

In the following, I will give you an overview of the most important metrics in the context of an example.

2.1. Terminology

To make the following metric definitions more intuitive to understand, let me first give an example. Consider running an application process. Because there are so many people applying, you want to use an ML model that does some prescreening, i.e., you want to build a model that predicts whether a candidate will succeed or fail on the job.

Formally, this is a binary classification problem, where the ML model f must predict a label Y_i for each candidate i, where $Y_i = 1$ indicates that an applicant is truly qualified for the role, and $Y_i = 0$ indicates that they are not. Let $Y_i \in \{0, 1\}$ indicate whether an applicant is truly qualified for the role, where $Y_i = 1$ means qualified, and $Y_i = 0$ means not qualified. To ensure a fair selection process, it is crucial to avoid making a prediction $\hat{Y}_i$ based solely on the **sensitive attributes** $A_i \subset X_i$ of candidate i, where X_i represents the set of attributes or features associated with candidate i. These sensitive attributes might include information such as gender or age. Now, consider the available data as $(X_i = x_i, Y_i = y_i)$. If we look at the model's prediction $\hat{y}_i = f(x_i) \in 0, 1$, we can assess the fairness of the model's predictions using fairness metrics. These metrics help evaluate whether the model's predictions are influenced by the sensitive attributes or if the model makes fair and unbiased predictions regardless of these attributes.

Based on this example, we will now look at some of the most important fairness metrics in more detail.

2.2. Group-based fairness metrics

In research, fairness metrics are typically classified into two main categories: **group-based fairness** and **individual fairness** (Barocas et al., 2017).

Group-based fairness metrics are very popular to quantify the fairness of a model because they are relatively easy to measure, and give key insights on model fairness (Caton & Haas, 2020). In the following, I outline the most important group-based metrics for you, but please be aware that there are many more. In particular, there are plenty of special metrics for particular use cases, e.g., a specific metric for the assessment of ranking systems (Zhu et al., 2020).

Statistical / Demographic parity:

This notion defines fairness as the probability of a given prediction being equal for all groups. In other words, statistical parity defines fairness as **independence between sensitive attributes and predictions**. In our example, this means that for all groups, the probability of being accepted for the job is the same, e.g., the group of men and women have the same chance to be accepted. Mathematically, demographic parity corresponds to:

$$P\left[\widehat{Y}_i = y \mid A_i = a\right] = P\left[\widehat{Y}_j = y \mid A_j = b\right], \qquad \forall y \in \{0,1\}.\, \forall a, b.\, \forall i \neq j.$$

Since probability distributions over the discrete domain of data points are relatively easy to measure, statistical parity is a straightforward way to assess the fairness of an ML model. At the same time, the metric has been criticized for being potentially unfair to individuals, because fairness at the group level does not guarantee fair treatment of all individuals (Dwork et al., 2012). For instance, when there is only a single qualified applicant from a certain group applied, the *base rate* in this group is very low, and adjusting your model for statistical parity may result in many false negatives in other groups with high base rates of qualified applicants. In fact, under different base rates, statistical parity and optimal classification are mutually exclusive (Garg et al., 2020).

Equal opportunity (Hardt et al., 2016):

Equal opportunity compares the *true-positive rates (TPR)* for the different groups in your data. In the context of fairness evaluation, the TPR measures the proportion of positive outcomes that are correctly identified by an ML model for a specific group or category:

$$TPR = \frac{number\ of\ true\ positives}{number\ of\ true\ positives + number\ of\ false\ negatives}$$

In other words, the TPR shows how well the model is capable of correctly identifying members of that group who actually belong to the positive outcome (e.g., qualified applicants). Fairness according to the equal opportunity metric is defined as the TPR being equal across all groups, which can be expressed as:

$$P\left[\widehat{Y}_i = y \mid A_i = a, Y_i = y\right] = P\left[\widehat{Y}_j = y \mid A_j = b, Y_j = y\right], \qquad \forall y \in \{0,1\}.\, \forall a, b.\, \forall i \neq j.$$

Here the sensitive attributes A do not influence the model predictions $\hat{Y}$. While this metric may deliver some valuable insights, it has been criticized in the past: As the TPR depends on the ground truth data, which may not be sufficiently unbiased itself, the equal opportunity metric may fail to identify unfair decisions (Barocas et al., 2017). Furthermore, by solely focusing on the TPR, this metric cannot capture the nature of all unfair decisions, e.g., consider what would happen if more unqualified applicants from one group than from the other are accepted (this would be reflected in the *false-positive rate; FPR*; defined analogously to the TPR).

Equalized odds (Hardt et al., 2016)**:**

This metric alleviates some of the issues of equal opportunity by imposing the additional constraint of equal FPR across groups. In other words, equalized odds is satisfied if and only if model prediction $\hat{Y}$ and sensitive attributes A are independent conditional on the true label Y, a property that is called **separation** (Barocas et al., 2017). In our binary example, binary case when:

$$\begin{aligned} P\left[\hat{Y}_i = y \mid A_i = a, Y = y\right] &= P\left[\hat{Y}_j = y \mid A_j = b, Y_j = y\right], \\ P\left[\hat{Y}_i \neq y \mid A_i = a, Y = y\right] &= P\left[\hat{Y}_j \neq y \mid A_j = b, Y_j = y\right] \end{aligned} \quad \forall y \in \{0,1\}.\, \forall a, b.\, \forall i \neq j.$$

This metric addresses the aforementioned flaw of the equal opportunity criterion, by not only looking at the TPR, but also the FPR of the model. Still, it remains a problem that this fairness score is only as good as the available ground truth test data. If this data itself is biased, then fully equalized odds may not tell the whole story (Barocas et al., 2017). Despite this issue, the equal odds metric has proven its value in the famous example of the COMPAS algorithm that was used to predict the recidivism rate of criminal offenders. While the algorithm did predict equal TPR for Caucasians and African Americans, the FPR was almost twice as high for African Americans, resulting in several unjust sentences. While the equal opportunity metric cannot capture this unfair treatment, the equality of odds does so.

Predictive rate parity (Chouldechova, 2017)**:**

Predictive rate parity is achieved if the probability of a positive outcome given a positive prediction (if you are a statistician, you will know this under the name of positive predictive value) is equal across groups. This criterion is equivalent to requiring independence between Y and A, conditioned on $\hat{Y}$, which is also called **sufficiency** or **calibration** (Barocas et al., 2017). Intuitively speaking, this means that the proportion of positive predictions relative to the proportion of positive examples should be the same across all groups (Dawid, 1982; Caton & Haas, 2020). Mathematically speaking, this criterion can be formulated as:

$$\begin{aligned} P\left[Y_i = y \mid A_i = a, \hat{Y} = y\right] &= P\left[Y_j = y \mid A_j = b, \hat{Y}_j = y\right], \\ P\left[Y_i \neq y \mid A_i = a, \hat{Y} = y\right] &= P\left[Y_j \neq y \mid A_j = b, \hat{Y}_j = y\right] \end{aligned} \quad \forall y \in \{0,1\}.\, \forall a, b.\, \forall i \neq j.$$

In our example, predictive rate parity implies that, for privileged and unprivileged people, the probability of an applicant with a good predicted qualification score being competent should be the same. Figure 2 illustrates what happens when this criterion is violated: if we hired any

applicants with a classifier score ≥ 0.6, we would reject all applicants from unprivileged backgrounds, despite their high qualifications for the job. This example illustrates why predictive rate parity is an important fairness criterion. A further benefit of using this metric is that it is compatible with a classifier being optimal, i.e., if $y = \hat{y}$, predictive parity follows directly (Barocas et al., 2017).

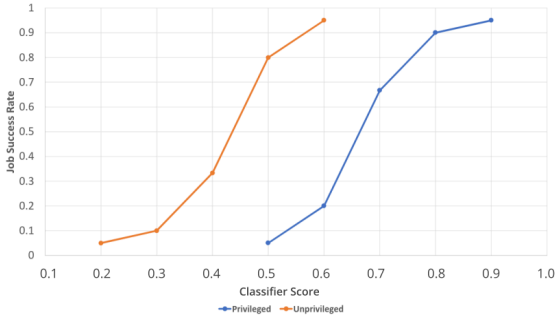

Figure 2: **The predictor for job qualification in this dummy example violates the predictive rate parity fairness criterion. Adapted from** Pessach & Shmueli (2022)**.**

2.3. Individual fairness

Contrary to the notions that emphasize equal treatment of different groups, individual fairness metrics try to reflect unfair treatment of individual persons. Given a reliable distance metric that quantifies the similarity between two individuals (Dwork et al.), individual fairness is typically defined as:

$$\left|P\left[\hat{Y}_i = y \mid X_i \backslash A_i\right] - P\left[\hat{Y}_j = y \mid X_j \backslash A_j\right]\right| < \epsilon, \qquad \forall y. \forall i \neq j. \quad d(X_i, X_j) \leq \epsilon$$

where $d(X_i, X_j)$ is a suitable distance metric that quantifies the similarity between candidates i, and j based on their non-sensitive attributes. The fundamental idea is that **individuals who have similar non-sensitive attributes or characteristics should receive similar predictions from the model**. This concept is illustrated in Figure 3. Picking similar data points to evaluate this metric can be tricky, especially when there is a lot of data. Thus, most research in this direction focuses on how to identify similar individuals. While it may be difficult to define a suitable distance metric (Pessach & Shmueli), the individual fairness notion can guarantee fairness between all individuals, and not only groups. In particular, since the fairness metric considers all attributes of an individual and not only group membership, it can be of great value (Dwork et al.).

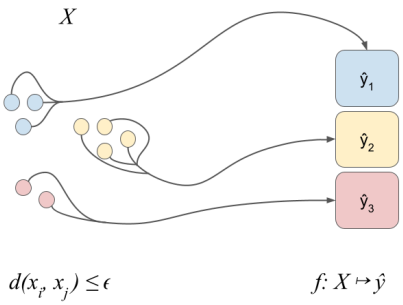

Figure 3: **Individual fairness intuition: All similar individuals should be treated similarly, i.e., similar inputs should lead to similar predictions. Note that the idea is that similarity is defined along all attributes, except for the sensitive ones.**

2.4. Why metrics do not tell the whole story about fairness

To make statements about the fairness of a specific model or algorithm, fairness metrics are indispensable. Yet, these metrics do not always tell the whole story about true fairness. Above, I have already stated some shortcomings of some of the above metrics, but there are even more fundamental issues.

First, some of these notions of fairness are mutually exclusive (Pessach & Shmueli, 2022). This result has been coined the **impossibility theorem**. For instance, it can be shown that *equalized odds* and *demographic parity* cannot be satisfied simultaneously in every scenario (Barocas et al., 2017). This mutual exclusivity may lead to striking unfairness. For instance, consider an application scenario in which there are many more qualified applicants from one group than the other. If we have a classifier that can classify all applicants perfectly, and all qualified applicants are hired, the FPR and TPR rates will be equal across groups (TPR=1, FPR=0), so equality of odds is satisfied. However, the demographic parity criterion will be violated, because application success and group membership are strongly correlated. So is our classifier fair in this scenario? This depends on your values and thus cannot be answered universally.

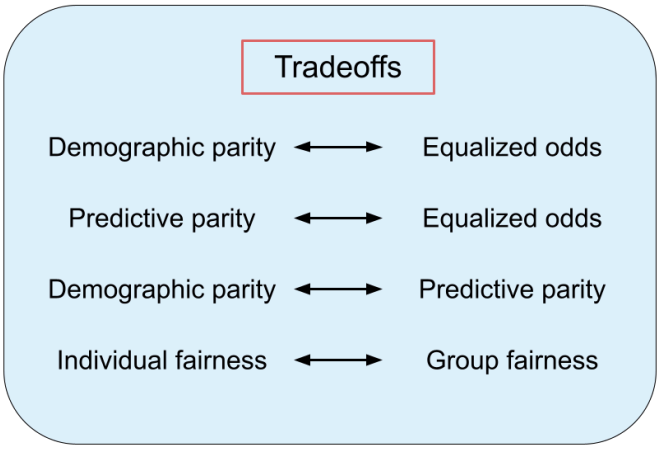

Figure 4: **The tradeoffs between the different fairness criteria. Adapted from** Pessach & Shmueli (2022)**.**

Fig. 4 gives an overview of the tradeoffs that need to be made in the binary classification setting. The **impossibility theorem** is very important in practice because it forces you to think thoroughly about which metric(s) you want to use. In particular, it means that satisfying a given notion of fairness does not guarantee completely fair decisions and that violating a given fairness metric may not be something that you would want to avoid at all costs.

I hope it is clear by now that all fairness metrics have advantages and disadvantages, and that you cannot select all of them to get the most out of fairness testing. Which metrics you should use for fairness testing will depend on the task, you must remember that your metrics are likely to not represent all aspects of fairness equally well. To get a hands-on idea

of the different strengths and weaknesses of these notions, I highly recommend this interactive demonstration.

---

3. Common pitfalls during model development

Now that you know how the fairness of an ML model can be assessed, you may be in a scenario where you find that the fairness of your model should be improved. But how to do this? Before, I will outline some things that you *absolutely must* avoid during development.

3.1. Fairness through unawareness

The first thing that comes to the mind of most people is to try and remove sensitive attributes to make sure that models cannot use them to make discriminatory decisions. This practice of ignoring sensitive attributes is often called fairness through unawareness, and it might just be a terrible idea. In the first place, unawareness by removing sensitive attributes is unlikely to make the outcome fairer, as there may be plenty of other attributes that are not explicitly sensitive, but can act as proxies for sensitive attributes, which means that they are highly correlated with the latter. Hence, solely removing the explicitly identified sensitive attributes might be insufficient to achieve fairness. In addition, you may in fact make your model even more likely to make unfair decisions without you noticing, because information on group membership is lost when sensitive attributes are removed (Kleinberg et al., 2018).

This is what happened to Amazon in the famous example of selecting regions to test their same-day delivery service: Based on consumer behavior predictions, Amazon ended up offering this service mostly in neighborhoods that are populated by a Caucasian majority (see Fig. 5). When Bloomberg called the company out on this, there was major public outrage and Amazon quickly changed their rollout strategy. This mistake could have been avoided if the model developers had used this information to avoid unfair decisions. Thus, it should always be avoided to simply ignore sensitive attributes to make model decisions fairer.

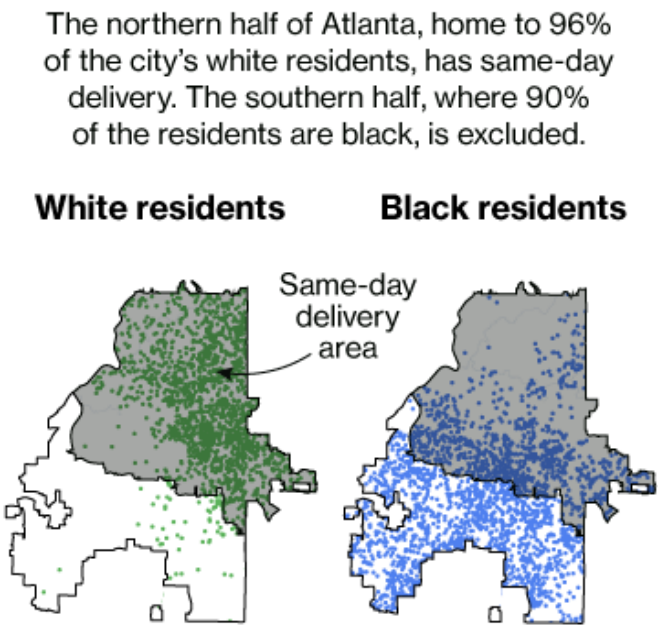

Figure 5: **The Amazon same-day delivery rollout excluded many areas primarily populated by black residents from access to their service, as this example from Atlanta shows. Source:** Bloomberg**.**

3.2. Focus on only fairness

A caveat of approaches to improving fairness is that *typically* model performance decreases with an increase in model fairness (Zafar et al., 2017; Kamiran & Calders, 2012). This case is exemplified in the extreme case of a model that outputs the same prediction for every input. Of course, this model is fair, but the model accuracy is at guessing level. For a theoretical examination of this tradeoff, I refer you to the analysis of Kamiran & Calders (2012) and the work of Liu & Vicente (2022).

This **accuracy/fairness tradeoff** implies that focusing on fairness alone may not be the best idea in all use cases. For instance, when predicting the recidivism of criminals, there is a tradeoff between accuracy, i.e., the safety of the society, and fairness for all parties involved (Corbett-Davies et al., 2017). In this case, it might be apt to keep the model accuracy as high as possible. In other scenarios, this might not be the case.

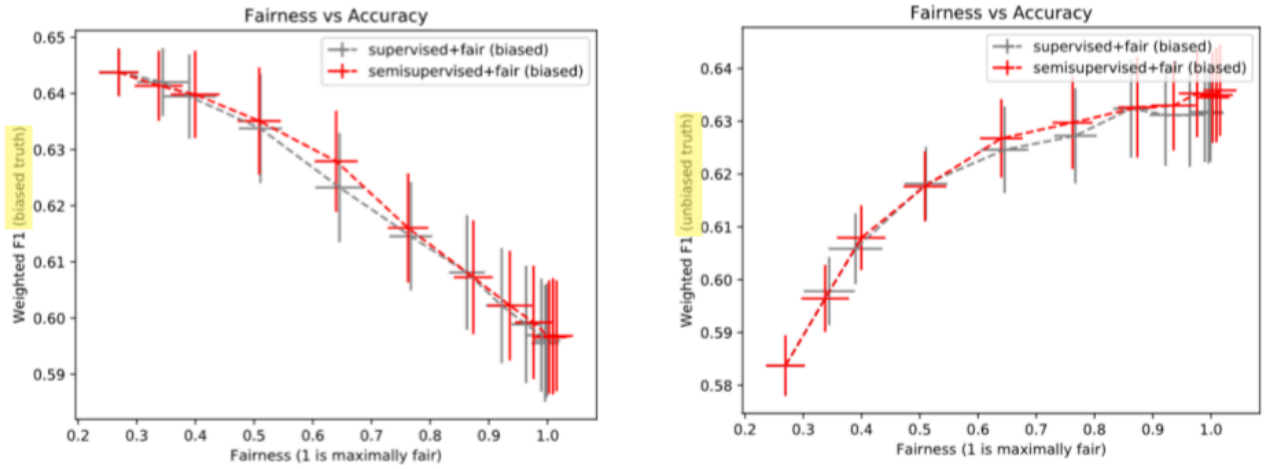

Figure 6: **Fairness accuracy tradeoff. Debiasing the ground truth alleviates the tradeoff and makes fairness and accuracy compatible. Source:** Wick et al. (2019)**.**

Before you are discouraged, note that this is not a *theorem* in the mathematical sense, because accuracy and fairness are not always mutually exclusive (hence I wrote *typically* in the paragraph above). In fact, Corbett-Davies et al. (2017) noted that by collecting more data, the classifier accuracy can be improved without decreasing fairness. Wick et al. (2019) further showed that debiasing the ground truth data may make accuracy and fairness compatible (see Fig. 6). Still, you should generally be aware that you may encounter situations in which you have to make a compromise concerning accuracy and fairness.

---

4. Methods to improve fairness in ML

Now that you know how the fairness of an ML model can be assessed, the main question that remains is how you can use these metrics to improve the fairness of your algorithm. This is still a hot topic of research, so there are many approaches (Barocas et al., 2017), which can be grouped as follows:

- **Pre-processing methods:** Transform your data such that fair predictions in any downstream task are possible.

- **In-processing methods:** Train a model that itself is fair according to your metric(s).

- **Post-processing methods:** Transform model outputs such that predictions are fair according to your metric(s).

4.1. Pre-processing methods:

Pre-processing methods generally try to address the problem of bias in the dataset (Caton & Haas, 2020). The goal of pre-processing methods is to adjust the training data distributions using transformations, such that the training data is no longer biased (Pessach & Shmueli, 2022). An advantage of pre-processing methods is that they only transform the

data, so they offer full freedom over the downstream task (Caton & Haas, 2020). This can for instance be beneficial in scenarios where a certain type of predictor shall be used.

Representation learning

A very common pre-processing method is **fair representation learning (FRL)**. The idea is to learn an embedding Z of the input data X, such that all correlations between sensitive features and outputs are removed, as illustrated in Fig. 7. One of the first FRL methods was presented by Zemel et al. (2013), who obtained fair representations by solving an optimization problem to maximize retained information (except for sensitive attributes) while minimizing the difference between the encodings of sensitive information. The resulting embedding is as similar as possible for all individuals that only differ on sensitive features. Algorithms following this idea are also implemented in fairness software packages, in case you want to use them in your project. Lahoti et al. (2019) presented another fair representation learning approach. Focusing on individual fairness, the idea of their method is to optimize a clustering of the data, such that all members of a given cluster are similar to each other, irrespective of their sensitive attributes. Consequentially, the cluster centroids are fair representations that can be used as classifier inputs. Another famous example of FRL is the fair variational autoencoder (VAE), which uses priors that enforce separation between the sensitive attributes and the latent encoding. For more works that use fair representations to alleviate fairness issues, I recommend you to look at the related work sections of the above papers, as well as the surveys by Caton & Haas (2020) and Pessach & Shmueli (2022).

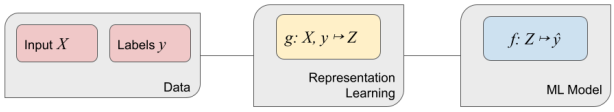

Figure 7: **The concept of representation learning for fairness. Adapted from** Cisse & Koyejo (2019)**.**

Resampling

A straightforward way to reduce biases in the dataset is **resampling** (Pessach & Shmueli, 2022). The idea behind (re-)sampling is to draw samples from the population, such that they are fair with respect to all sensitive attributes, while simultaneously depicting a representative range of attribute values (Caton & Haas, 2020). A classic resampling technique, inspired by learning theory, was conceived by Kamiran & Calders (2012), who observed that data points near the decision boundary are the most likely to be subject to unfair classification. To alleviate this issue, the authors proposed using a preliminary classifier to determine a reasonable decision boundary, then sample data points close to this boundary using bootstrapping. More recently, Chakraborty et al. (2021) presented the Fair-SMOTE algorithm that synthetically generates new samples, such that all sensitive subgroups are represented equally, and subsequently deletes all biased samples from this data set using a technique called situation testing. Closely related to sampling-based fairness methods are the notions of **geometric diversity** and **combinatorial diversity**, which can be used to measure the diversity of a given dataset. It has been shown that maximizing these quantities, for instance via determinantal point process-based sampling

(DPPs), can increase the fairness of the resulting ML model decisions (Celis et al., 2016). Of course, it will not be possible to sample or simulate new data in all practical scenarios, but if this is possible, resampling is a very effective method to improve fairness (e.g. An et al., 2021; Romano et al., 2020).

Reweighing

Reweighing methods avoid the sampling of new data by assigning weights to existing samples, such that the resulting dataset is less biased. With this goal in mind, Kamiran & Calders (2012), for instance, proposed to weigh each data point by the following weight:

$$w_i(a_i, y_i) = \frac{P_{expected}[a_i, y_i]}{P_{observed}[a_i, y_i]} = \frac{P[x_i \mid a_i]P[x_i \mid y_i]}{P[x_i \mid a_i, y_i]}$$

In other words, every instance is assigned a weight that adjusts for the difference between the expected and the observed probability of being assigned a given outcome label. This way, the weights compensate for the different base rates in the population, and y and a become independent. To estimate the conditionals in the above equation, it suffices to count the occurrences of attribute-label combinations in the dataset when performing classification. A benefit of using reweighing methods is that potentially difficult sampling is avoided, while fair treatment can be guaranteed (Caton & Haas, 2020).

Other methods

Other notable pre-processing include **relabelling** approaches that try to adjust the value of the dependent variable (Caton & Haas, 2020), and **causal methods**, which use causal models to guarantee that fairness constraints are met, e.g., by providing an understanding of which attributes *really* cause unfair predictions and should be removed (Kusner et al., 2017; Chiappa & Isaac, 2019). For more detailed information, I recommend you to look at the excellent survey of Caton & Haas (2020).

4.2. In-processing methods

In-processing methods have the goal to modify the actual ML model, such that the resulting predictions do not discriminate against anyone. An advantage of in-processing methods is that they offer full control over the accuracy vs. fairness tradeoff, and thus typically tend to score well on both accuracy and fairness (Caton & Haas, 2020).

Improving model generalization

An elegant approach to combat biased model predictions is to improve the model's generalization and robustness capabilities. This approach is based on the work of Lan & Huan (2017), who noted that a major source of model unfairness is distribution shift. From this, it follows that developing methods, which are robust to distribution shifts, do not only perform better in terms of accuracy but also in terms of fairness. Potential remedies to this issue include general techniques for increasing model robustness, for instance by using specific data augmentations (Qin et al., 2022; Zhong et al., 2020; Rebuffi et al., 2021). Other works explicitly address the fairness transfer across distributions by using adversarial learning to transfer the fairness properties of models across data distributions (Madras et

al., 2018; Schumann et al., 2019). I think that the generalization-based approach is generally very elegant because you simultaneously optimize model performance and fairness.

Fair optimization

Fair optimization methods are methods that improve model fairness by integrating it into the optimization problem that is typically solved to train models. The most straightforward way to do this is by adding one or several **regularisation** terms to the optimization objective to enforce fair results (Berk et al., 2017; Beutel et al., 2019; Kamishima et al., 2011). This approach has the benefit that the model architecture and optimization procedure can remain unchanged. Furthermore, it permits to "tune in" fairness, via a parameter λ (similar to regularization in ridge regression). The idea of these regularization-based approaches is illustrated in the following expression, where $Fairness(\cdot, \cdot) \mapsto (0, 1))$ is a function that measures the fairness of a given prediction f, given model weights w:

$$\min_{w} \quad Loss(y_i, f(x_i, w)) + \lambda [1 - Fairness(y_i, f(x_i, w))]$$

Since fairness is essentially acting as a model regulariser in this objective, the model accuracy will likely be lowered when using such approaches (Zemel et al., 2013). Alternatively, fairness can be enforced by adding **fairness constraints** to the optimization problem (Perrone et al., 2021; Donini et al., 2018; Komiyama et al., 2018). Constrained methods have the major benefit of *guaranteeing* a fair solution. However, constrained optimization problems are typically more difficult to solve, so the optimization procedure has to be adapted.

Explainable AI

Although not directly optimizing fairness, explainable ML methods can greatly improve the fairness of ML models indirectly (Zhou et al., 2022). By offering an understanding of which attributes are used during prediction, unfair predictions can be detected faster and potentially be fixed better. Ideally, explainability methods for ML can indicate where your model is unfair, so you can adjust it during development. Thus, it may be worth thinking about using methods that can be used in combination with explainability tools such as Shap or Lime. For instance, Google's What-If tool can help you to visualize the model predictions per group, which in turn can be used to get an understanding of prediction fairness. This is illustrated in Figure 8.

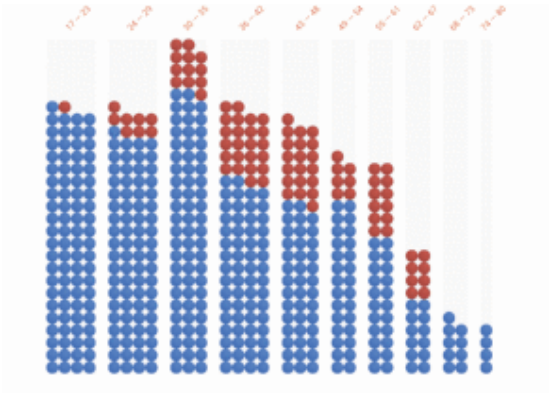

Figure 8: **Google's what-if tool can unveil different predictive rates between groups. Taken from** Wexler et al. (2020)**.**

Adversarial learning

Strictly speaking, more a family of learning algorithms than a class of fairness-inducing methods, adversarial learning has been increasingly popular to improve model fairness in the past five years. Beutel et al. (2017) trained two adversarial networks to maximize the predictor's capability to accurately predict the output classes, while simultaneously learning data representations that minimize the discriminator's capability to predict the sensitive feature. Celis & Keswani (2019) solved a similar min-max optimization problem, but instead of tweaking the data representation in the update step, they update the classifier directly. For other papers using adversarial learning, I recommend you to look at the survey of Pessach & Shmueli (2022).

Other methods

The above methods are quite general and do not necessarily impose constraints on your model architecture. While this can be seen as a strength, you might encounter a situation in which you may want to use fairness methods tailored to your use case. For this purpose, there is an increasing body of research that focuses on **fairness for specific tasks**, such as fair word embeddings for natural language processing (NLP), fair clustering, or fair recommender systems. Recently, the work in this area was summarised in an interesting review paper that I highly recommend.

4.3. Post-processing methods

Post-processing methods transform the posterior distribution of a model such that fairness criteria are satisfied, which is also called **posterior editing** (Barocas et al., 2017). Like pre-processing methods, post-processing methods can be used for any prediction task and thus offer nice flexibility properties, e.g., they are compatible with any model (Caton & Haas, 2020). Furthermore, post-processing methods are attractive, because they achieve fairness without necessitating model retraining (Petersen et al., 2021).

Thresholding

Thresholding methods try to find classification thresholds for each group, such that fairness constraints are met (Caton & Haas, 2020). The classic example of a thresholding method is from Hardt et al. (2016) Their approach enables to produce predictions that satisfy the equal opportunity and equalized odds fairness criteria. The algorithm works by adjusting the decision thresholds of the classifier based on the observed TPR and FPR. To determine the exact thresholds, an optimization problem is solved that finds decision thresholds that minimize the loss under the equality of opportunity (or odds) constraint. For binary classification, the solution of the equal odds optimization problem has a nice geometrical interpretation, where the optimal solution corresponds to the intersection of the ROC curves for both sensitive groups, as depicted in Fig. 9. The algorithm is very popular to this day, being implemented in fairness tools such as Fairlearn and AIF360.

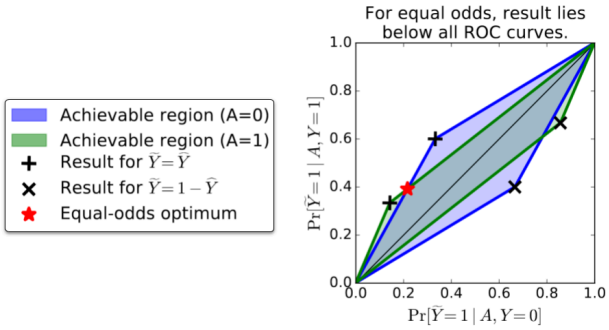

Figure 9: **The geometric intuition of Hardt et al.'s approach to fairness by thresholding. For equal odds, the optimum lies at the intersection of the ROC curves of the two sensitive groups $A = 0$ and $A = 1$. Taken from** Hardt et al. (2016)**.**

In a similar vein, Fish et al. (2016) presented an algorithm, which accepts as input the prediction confidence of a classifier (e.g. SVM or boosted tree ensembles) and finds the minimal error decision boundary shift that achieves demographic parity, by flipping the output labels of data points with small confidence. The result is a computationally lightweight algorithm that offers control over the model fairness vs. accuracy tradeoff *after* model training.

For further thresholding methods that use different optimization problems or different solution techniques, I recommend you to look at the papers of Kamiran et al. (2012), Corbett-Davies et al. (2017), and Alabdulmohsin & Lucic (2021).

Calibration

I introduced the notion of classifier calibration when I introduced the predictive rate parity fairness criterion, noting that the two are used as equivalent terms (Barocas et al., 2017). Pleiss et al. (2017) noted that post-processing methods that achieve equal odds, such as Hardt et al.'s approach, often produce uncalibrated classifiers that violate the predictive rate parity. The authors demonstrate that there is a tradeoff between calibration and equal odds. To alleviate this issue, the authors presented an algorithm that optimizes over calibrated classifier outputs to find probabilities with output labels that should be flipped, to maximize the equality of odds. Although calibration methods may work in certain scenarios and have been incorporated into the AIF360 toolkit the authors note that depending on the problem it might be better to perform *either* calibration or thresholding for equal odds.

Other methods

Since thresholding only works in the context of classification, there have also been developed other post-processing methods for other settings. For instance, Johnson et al. (2016) proposed an approach to correct regression outputs a-posteriori, Nandy et al. (2022) presented an approach for fairness improving ranking adjustments, and Wei et al. (2020) developed a method to adjust predicted probability scores, as they are common in ML.

---

5. Best practices

We have seen that fairness has become a popular topic and that there are multiple approaches in the research on how model fairness can be assessed and improved. Now I will come to the last, and maybe most important part of this article: What can you do as a practitioner?

This question is key because it is the practitioners who engineer the ML systems that affect people's lives. Thus practitioners should follow best practices that help to prevent their models from producing discriminating outputs. In this section, I gathered advice from academia and software development (e.g., Google; UC Berkeley; Barocas et al., 2017), which can help you to build better models and software.

Examine your data

Although this may sound trivial, this may be the most important practice towards fair ML. Before you start developing a model, you should have a well-posed problem definition that specifies who will end up using your product, so that you can make sure that your data represents this population well enough. Once you have collected a representative dataset, you then must verify that this data does not contain any bias. To check for biases, you can use some of the tools that I present at the end of this article, for instance, Aequitas or Know your data.

Be aware of the purposes and limitations of your model

No model is perfect for everything. You should always keep this in mind when developing and deploying ML models. This means that you should not overestimate what your model can do, e.g., if your model can detect correlations you should not use it for causal predictions. Being aware of the limitations of your model also includes being aware of the shortcomings of ML in general. As Ruha Benjamin (2019) pointed out, mathematical models may seem objective and neutral, but at each step of your model development, your design choices affect the resulting model, which will be reflected in its predictions. This should be kept in mind whenever an ML model is deployed in real-world settings.

Test your model (multiple times)

If you develop machine learning models that will be used in the real world, you must test your models. This holds for fairness just as much as for accuracy. Proper testing means that you should:

- Evaluate your model across multiple classification thresholds, to gain an understanding of how different thresholds affect different groups.
- Test on multiple datasets, e.g., use this dataset to additionally test a face recognition system to make sure that you can correctly classify people with different appearances.
- Test your model *before*, but also *during* deployment to check for any unfair decisions due to distribution shifts. In case issues arise during deployment, you should have a clear roadmap of what to do.

To perform model tests, there are multiple tools that you can use, such as the Tensorflow fairness indicators that can be integrated into your Vertex AI pipeline, or the Fairlearn package if you want to go full open-source.

Choose the right metrics

As emphasized above, there are tradeoffs between the different metrics, which makes metric selection non-trivial. Thus, the appropriate fairness metric must be chosen depending on the context. Despite the aforementioned tradeoffs, some notions may also be combined in certain use cases. When selecting a fairness metric, ask yourself what each metric measures. For instance, ask yourself whether it makes sense to use a ground truth-based metric (e.g. equalized odds) with the data you have at hand, or whether you should rather focus on calibration.

Report your results

Model cards have been introduced to serve as structured summaries of ML model properties. These cards may contain information about the data distribution that was used for model training, or the environmental impact of your model training. Providing such a model card may help to make your development process more transparent, and can help to detect fairness issues faster.

Use tools

As of 2023, there is a wide range of tools that can help you to make sure that your model is fair. These tools are designed to either detect, and/or mitigate discrimination and bias in your model. There exist multiple overviews of fairness tools (for instance here, here, and here). To me, the most important ones are (in alphabetical order):

- **Aequitas:** This open-source bias audit toolkit provides you with a Python library to measure the fairness metrics of your data and model. Additionally, the toolkit provides a CLI that can be used to audit your data and model. This step can conveniently be integrated into a CI/CD pipeline, so you can continuously check for bias and fairness.
- **AI Fairness 360 toolkit (AIF360):** This Python toolkit can be used to implement some of the pre- and post-processing methods that I outlined above, such as posterior editing. Support for multi-class classification problems, metrics for individual fairness, and a GUI make the AIF360 toolkit quite useful.
- **Fairlearn:** Another Python package with metrics to measure fairness and tools to improve it. Initially launched by Microsoft, it is now fully community-driven. Alongside classification problem support, you can also test for fairness in regression settings, which is possible with some other tools.
- **Tensorflow fairness indicators:** This Python library enables you to compute the most important fairness metrics on your data in binary and multi-class classification tasks. It integrates nicely with Tensorflow, and there exist plugins for Tensorboard and other libraries.
- **Know your data:** This dashboard-based tool by Google focuses on the dataset aspect of fairness. With this tool, you can examine your data to identify biases and other issues in your data before you go into production. Some of the tool's features are also included in Google's Vertex AI for MLOps.
- **What-If Tool:** This tool enables you to visualize the decisions of your model in multiple settings, such that you can gain an understanding of how it is working. For instance, you can test how the model predictions vary, if a certain feature is kept constant. There is a wide range of visualizations that you can choose, e.g., the one in Fig. 8. The

What-If tool is very appealing because you can gain a qualitative understanding of your model performance.

Disclaimer: This list is not comprehensive, and reflects my personal opinion as of May 2023. Please be aware that all of these tools come with limitations.

Stand on the shoulders of giants

There is much more to fairness in ML than what I can cover in a brief overview article such as this. There is this excellent MLSS talk by Moritz Hardt. There are multiple books such as Cathy O'Neil's Weapons of Math Destruction (2016), Virginia Eubanks's Automating Inequality (2018), and Ruha Benjamin's Race After Technology (2019). I already cited existing survey papers throughout this article, and I again highly recommend you the works of Caton & Haas (2020), Pessach & Shmueli (2022), and Barocas et al. (2017). Beyond mathematics and computer science, there is even more work on fairness in many other disciplines. These disciplines have been interested in fairness for much longer, so it may be useful to draw some inspiration from sociology or law. Thus, always remember to *stand on the shoulders of giants*.

---

6. Summary

With the continued rise in the usage of such models, guaranteeing the fairness of model decisions is more important than ever. In this post, I outlined the most important metrics that can be used to monitor the fairness of model predictions, and important techniques to improve model fairness, namely via pre-, in-, and post-processing methods. Finally, I listed some tools and software that you can use as a practitioner. I hope that this article showed you that fairness is a very important topic in ML nowadays and that there are ways to improve the fairness of your model without needing to change too much of your workflow.

Demos
Blog
Recorded Talks
Industries
Newsletter

Company

About us
Company News
Jobs
Publication Archive
Contact

dida Datenschmiede GmbH
Email: info@dida.do
Phone: +49 30 921058800
Hauptstr. 8 (Meisenbach Höfe,
Aufgang 3a), 10827 Berlin,
Germany