ISYE 8813: Algorithmic Foundations of Ethical ML **Fall 2021**

Lectures 19: Between Individual Fairness and Group Fairness

Lecturer: Juba Ziani

In the previous lectures, we saw some of the shortcomings of statistical fairness. Namely, while it provides intuitive fairness guarantees at the level of the groups we define, it generally does not provide any guarantee at the level of individuals in each of these groups, or even across smaller sub-groups of these groups. However, individual fairness also has shortcomings, in that it is hard to satisfy, for the following reasons:

1. **Specifying the metric** One of the most challenging aspects of individual fairness is to specify the metric d that we are going to use. In particular, we have to decide when two individuals x and x' are close enough to each other that we should make similar decisions on them.

   One common approach in the literature to decide on such a metric is to try to infer it from human auditors, that may not be able to specify the metric themselves, but have some ability to tell you whether they believe two individuals x and x' should be treated the same way.

   One approach taken in related work is to try to learn d directly from such observations. This is, however, generally a hard learning problem: how can I learn a metric $d(x, x')$ in a high-dimensional space V? Imagine I have sample access to i.i.d. pairs (x, x') of features: even if $V = 0, 1^d$, I would need exponentially many samples to able to learn the metric (enough samples for each possible (x, x') pair to learn $d(x, x')$ accurately). If V is not a discrete set, it will be impossible to cover the infinitely many possible (x, x') pairs, and I will need to make more assumptions about V or d to deal with such difficulties.

2. **Learning on singletons is impossible** Many specializations of individual fairness look at the feature x of an agent being actually this agent's true label, i.e. $x \triangleq y$. In this case, we are now requiring that two agents with similar labels $y \sim y'$ have similar distributions of outcomes $M(y) \sim M(y')$.

   But this is a statistically impossible task! The reason we are able to do so for group fairness is that we are allowed to average on the many individuals that constitute a group; we can use the law of large number to understand properties of $\Pr[Y = 1|X]$ at the group level. But this becomes impossible at the level of a single individual, making such a statistical notion of individual fairness impossible to satisfy in practice. It is a lot easier to predict roughly how many people in a large group of similar individuals are going to commit a crime, rather than identifying exactly which individuals in this group are likely to or actually going to commit a crime.

In turn, we may want to look at definitions of fairness that are in-between, and satisfy the best of both worlds. It may be hard to have fairness hold at the individual level for

the reasons described above, but it may be possible instead to have fairness hold across *all specified sub-groups*, rather than only across the two groups we have initially defined. Note here that we are allowing overlap across subgroups too!

See for example Figure 1 below, which shows a typical example of *fairness gerrymandering*. So here, imagine we have two possible divisions of our population in groups: by gender, so male/female (limiting ourselves to binary genders for simplicity only), or by race, say black or white; note that these two ways of partitioning the whole population overlap with each other. Imagine we want to make accept/reject decisions, and we want demographic parity across gender. The solution in the figure provides such demographic parity across gender: there are the same number of male green dots and female green dots. In fact, it also provides demographic parity over ethnicity. Hence, here, we appear to obtain fairness when we look at a single sensitive attribute at a time. But the picture is different when we look at *conjunction* of sensitive attributes: here, we are obviously unfair towards white females and black males. I.e., now that we are looking at *sub-groups* defined by a *combination* of several sensitive attributes here, our notion of statistical parity does not hold anymore! Even worse, we can make it arbitrarily bad against whichever sub-group we want.

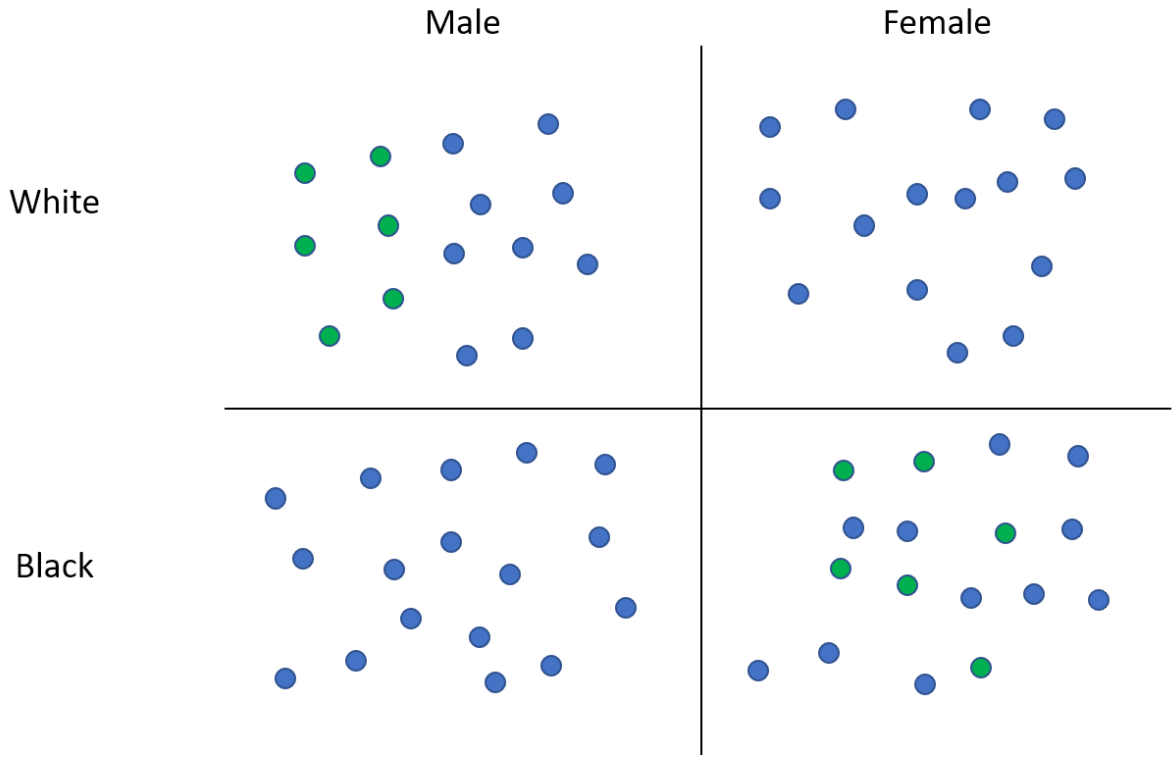

Figure 1: Fairness Gerrymandering. Agents with outcome $R = 1$ are in green, the remaining agents in blue.

We call this fairness *gerrymandering*. This follows directly from the term and the idea of gerrymandering in politics/elections: how I define my sub-groups (in the case of elections,

districts or states) has an influence on the outcome of the machine learning algorithm (here, election). I can effectively manipulate the vote of part of a political party by defining state/district lines such that the this political party is a minority in most states, but a majority in a few states. Then, some of the sub-groups defined by such (political party, state) pairs may be treated unfairly (even though there is fairness across political parties in the sense that the popular vote is proportional to the number of voters in each party, and even in an idea world in which we are fair across states in the sense that the weight of the vote of each state is proportional to the size of the state).

We will look today at two papers that aim to address such issues, by guaranteeing fairness across all sub-groups (possibly overlapping) that we specify: such sub-groups may be defined as a combination of several sensitive attribute. We are in some way "interpolating" now between individual and group fairness: while we do not go all the way towards individual fairness, we still considers fairness beyond simply a single sensitive attribute. We now have a notion of fairness that is still statistical, but is possibly much more fine-grained than before.

1 Preventing Fairness Gerrymandering: Statistical Parity and Equal Opportunity

This part of the lecture is based on [KNRW18]. There, the idea is to guarantee this kind of fairness across sub-groups for the first two types of notions of statistical fairness we saw in class: statistical parity, and equality of true positive and/or false positive rates.

1.1 Model and Objectives

Model Each individual is going to be described as a tuple $((x, x'), y)$. $x =$ sensitive/protected attributes, $x' =$ non-sensitive/unprotected attributes, and y is as before the true label. Note now that x does not have to be a single sensitive attribute: x can be a *vector* constituted of several sensitive attributes. The individuals are drawn i.i.d. from a joint distribution $\mathcal{P}$ over $((x, x'), y))$.

We restrict ourselves to the binary setting, once again, where $y \in \{0, 1\}$. We write $X = (x, x')$ to denote the joint feature vector here, and make decisions $D(X) \in \{0, 1\}$, also restricted to be binary (but D may be randomized). We now want to be able to protect *sub-groups*, that in particular may be defined by several combinations of protected attributes (rather than by a single protected attributes). To define these sub-groups, we introduce a function g for each protected subgroup G: g defines the sub-group, with $g(x) \triangleq 1$ if and only if $x \in G$. Note that we will use g and G interchangeably in this section.

Finally, we let $\mathcal{G}$ denote the set of all sub-groups (or equivalently, of all functions g) we consider. $\mathcal{G}$ is then the set of sub-groups across which we aim to guarantee statistical fairness. Note that the sub-groups may overlap, and that individuals may belong to several sub-groups. For example, in our example of Figure 1, we could define, letting M/F the subsets of male/female individuals, and B/W the subsets of black/white individuals.:

- $\mathcal{G} = \{M, F\}$, in which case we just want group fairness across the gender attribute.
- $\mathcal{G} = \{B, W\}$, in which case we just want group fairness across the race attribute.
- $\mathcal{G} = \{B, W, M, F\}$. Note that the example above satisfies this: the number of white, black, male, and female people that we accept is the same (5 every time).
- $\mathcal{G} = \{B \cap F, W \cap F\}$, in which case we want fairness across black females and white females. Now we are starting to look at groups that are specified by a combination of several protected attributes!
- $\mathcal{G} = \{B, W, B \cap F\}$, in which case we want fairness across black and white people, but also black females. In this case note that the groups are overlapping.
- $\mathcal{G} = \{B, W, M, F, B \cap F, W \cap F, B \cap M, W \cap M\}$ in which we want fairness with respect to every possible collection of one or two protected attributes.

Our definitions of fairness We are interested here in the two following definitions of fairness:

Definition 1 (Statistical Parity)**.** *We satisfy γ-statistical (SP) fairness if and only if, for **all** $g \in \mathcal{G}$,*

$$\Pr[g(x) = 1] \left|\Pr[D(X) = 1] - \Pr[D(X) = 1 | g(x) = 1]\right| \leq \gamma.$$

Informally, what the definitions above does is the following: it looks at all possible groups g, and requires that the decisions we make in this sub-group are close to the decision we make across the entire population. I.e., the distribution of $D(x)$ does not change (much) when conditioning on g and stays close to the "base" rate $\Pr[D(X) = 1]$ in the whole population. This effectively guarantees that g is roughly independent of $D(x)$, which is what we want for statistical parity.

We multiply by $\Pr[g(x) = 1]$ for two reasons. The first one is just so that we limit ourselves to the agents/protected feature vectors that belong in group G (we do not have to protect agents not in the current sub-group we consider, G). The second one is that this also implies here that we require easier and more lax guarantees of fairness for smaller groups: if G is small and $P[g(x) = 1]$ is hence small, the constraint above becomes easier to satisfy. A reason to do so is because statistically and from a learning perspective, such statistical properties become harder to satisfy when the group size is smaller, so we want the guarantees to become more permissive for smaller groups.

Definition 2 (False Positive (FP) Sub-Group Fairness)**.** *We satisfy γ-False Positive (FP) fairness if and only if, for **all** $g \in \mathcal{G}$,*

$$\Pr[g(x) = 1 \cap y = 0] \left|\Pr[D(X) = 1 | y = 0] - \Pr[D(X) = 1 | g(x) = 1 \cap y = 0]\right| \leq \gamma.$$

Informally, the idea is the same as for and symmetric to statistical parity, but we now take the label y into account, and only consider events in which $y = 0$. Among these events, we want roughly the same rate of the prediction being $D(X) = 1$ (i.e., a false positive) across all groups g, and this rate should roughly be the base rate $\Pr[D(X) = 1|y = 0]$ across the whole population of agents with negative labels.

1.2 Auditing Fairness

I will refer to the paper for more details on auditing for fairness. But the high-level idea here is to detect with high probability and computationally efficiently, from samples from the distribution $\mathcal{P}$, whether a given decision rule is γ-fair.

[KNRW18] shows that auditing for fairness is equivalent to "weak agnostic learning" when trying to learn to predict the label $D(X)$ simply from the protected features x (in the case of statistical parity, otherwise you want to learn $D(X)$ from x *conditional on* $y = 0$). Here, we are talking about "weak" learning because we only require to do slightly better than random guessing (i.e. we want a probability of error of at most $1/2 - \epsilon'$, in polynomial time in $1/\epsilon'$, for some small ϵ').

A few remarks:

- Intuitively, this makes sense. If you can find a classifier that predicts $D(X)$ well from the protected features x, you can audit or test how much the distribution of $D(X)$ varies across different x's, hence different group.
- The good news: there is something nice about the result, which is that you don't need to predict $D(X)$ "very" well: a weak notion of learning where you do slightly better than random guessing is enough!
- The bad news: weak agnostic learning is still known to be intractable for relatively simple classes of groups such as conjunctions of boolean attributes or linear threshold functions. (There are however good heuristics than we can use for weak agnostic learning, hence for auditing fairness).

1.3 Algorithms for Fairness

Here we focus on the statistical parity definition of fairness, but note that all techniques and results also hold for FP fairness. Our goal is the following: minimize the error of our classifier, while satisfying fairness. I will give a high-level overview of how the paper solves this problem, but will refer to [KNRW18] for some of the technical details.

The optimization program We imagine that we have an hypothesis class $\mathcal{H}$, and want to finds the best possible (randomized) hypothesis in this class that is fair. I.e., we are interested in finding a probability distribution D over $\mathcal{H}$ (we can denote the set of such probability distributions Δ_H) such that:

1. D has low expected error, where the error of an hypothesis h is

$$err(h, \mathcal{P}) = \Pr[h(x, x') \neq y],$$

   i.e., the probability of predicting the binary label wrong.

2. D is fair, in the sense that Definition 2 holds for all $g \in \mathcal{G}$, over distribution D.

For simplicity of notations, we let $\alpha(g, \mathcal{P}) \triangleq \Pr[g(x) = 1]$ and $\beta(g, D, \mathcal{P}) \triangleq |\Pr[D(X) = 1] - \Pr[D(X) = 1 | g(x) = 1]|$. Note that because our groups may be overlapping, we cannot treat them separately: the way we design $D(X)$ for a given protected attribute vector x may impact several groups at the same time! In fact, here, we are dealing with an *optimization* problem that has fairness constraints that can overlap and take the same protected feature vector into account several times. Our optimization problem can then be written as follows:

$$\begin{aligned} \text{Fair} = \min_{D \in \Delta_{\mathcal{H}}} \quad & \mathbb{E}_{h \sim D}[err(h, \mathcal{P})] \\ \text{s.t.} \quad & \alpha(g, \mathcal{P})\beta(g, D, \mathcal{P}) \leq \gamma \ \forall g \in \mathcal{G}, \end{aligned}$$

We aim to approximately optimize this given finitely many samples $(X_1, y_1), \ldots, (X_n, y_n)$ from data distribution $\mathcal{P}$. SInce we only have access to samples from $\mathcal{P}$, we instead write the optimization problem over the empirical distribution of the samples, that we call $\mathcal{P}_n$. We call the corresponding problem Fair ERM:

$$\begin{aligned} \text{Fair ERM} \triangleq \min_{D \in \Delta_{\mathcal{H}}} \quad & \mathbb{E}_{h \sim D}[err(h, \mathcal{P})] \\ \text{s.t.} \quad & \alpha(g, \mathcal{P})\beta(g, D, \mathcal{P}) \leq \gamma \ \forall g \in \mathcal{G}, \end{aligned}$$

Solving the optimization We do so in several steps:

1. Note that $\mathcal{H}$ and $\mathcal{G}$ might be infinite. In turn, we focus on hypothesis classes $\mathcal{H}$ and classes of sub-groups $\mathcal{G}$ that have bounded VC-dimension. We let $d_1 = VCDIM(\mathcal{H})$ and $d_2 = VCDIM(\mathcal{G})$. The reason to do so is that a hypothesis class is PAC learnable iff it has finite VC dimension, and there is no hope otherwise; we need finite VC dimension for $\mathcal{H}$ for learnability of the objective, and of $\mathcal{G}$ for learnability of the fairness constraint. We will use this finite VC-dimension property and Sauer's lemma to rewrite our optimization problem in a more tractable way:

   Lemma 3 (Sauer's Lemma)**.** *Let $S = (x_1, \ldots, x_n)$ be a data set of size n. We have $|\mathcal{H}(S)| = O(n^{d_1})$ and $|\mathcal{G}(S)| = O(n^{d_2})$, where $\mathcal{G}(S) = \{g(x_1), \ldots, g(x_n) | g \in \mathcal{G}\}$ and $\mathcal{H}(S) = \{h(x_1), \ldots, h(x_n) | g \in \mathcal{H}\}$.*

   I.e., there is at most $O(n^d)$ ways of labelling n data points with hypotheses in a class of VC-dimension at most d. So here, we can leverage the fact that i) hypotheses can be entirely described by how they label data points and ii) the fact that even though

the hypothesis class is infinite, there is only a finite number of possible labellings of our n sample points. We can replace $\mathcal{H}$ and $\mathcal{G}$ in our optimization problem by the finite numbers of hypotheses $\mathcal{H}(S)$ and groups $\mathcal{G}(S)$ defined over our n points, to get the following, now finite, optimization problem:

$$\begin{aligned}\text{Fair ERM} \triangleq \min_{D \in \Delta_{\mathcal{H}(S)}} \quad & \mathbb{E}_{h \sim D}\left[err(h, \mathcal{P}_n)\right] \\ \text{s.t.} \quad & \alpha(g, \mathcal{P}_n)\beta(g, D, \mathcal{P}_n) \leq \gamma \ \forall g \in \mathcal{G}(S),\end{aligned}$$

2. Now, we need to solve the finite optimization program we just developed. We can rewrite the problem immediately as the following (convex) optimization problem:

$$\begin{aligned}\text{Fair ERM} \triangleq \min_{D \in \Delta_{\mathcal{H}(S)}} \quad & \mathbb{E}_{h \sim D}\left[err(h, \mathcal{P}_n)\right] \\ \text{s.t.} \quad & \phi_+(D, g) \leq 0, \ \phi_-(D, g) \leq 0 \ \forall g \in \mathcal{G}(S),\end{aligned}$$

where

$$\begin{aligned}\phi^+(D, g) &= \alpha(g, \mathcal{P}_n)\left(\Pr\left[D(X) = 1\right] - \Pr\left[D(X) = 1 | g(x) = 1\right]\right) - \gamma, \\ \phi^-(D, g) &= \alpha(g, \mathcal{P}_n)\left(\Pr\left[D(X) = 1 | g(x) = 1\right] - \Pr\left[D(X) = 1\right]\right) - \gamma.\end{aligned}$$

It is not too hard to see that it is a convex optimization problem, by noting that we are optimizing on the following variables $\Pr[D = h]$ for all $h \in \mathcal{H}(S)$ (i.e. the probability we assign to each labelling), and replacing this explicitly in the optimization problem. But, note that we have n^{d_1} variables, and n^{d_2} fairness constraints. For d_1 and d_2 large enough, this could be an intractable problem. Below, we will show how to reduce this to a problem that is arguably simpler.

We can then write the Lagrangian

$$\mathcal{L}(D, \lambda) = \mathbb{E}_{h \sim D}\left[err(h, \mathcal{P}_n)\right] + \sum_{g \in \mathcal{G}(S)} \left(\lambda_g^+ \phi^+(D, g) + \lambda_g^- \phi_-(D, g)\right),$$

and note that by strong duality (we have a convex, finite optimization program with a non-trivial feasible region),

$$\text{Fair ERM} = \max_{\lambda \geq 0} \min_{D} \mathcal{L}(D, \lambda).$$

3. Note that now, the problem we are trying to solve is a *zero-sum game* between two players: the analyst/learner (us) trying to play a distribution D that minimizes the Lagrangian, and an adversary that is trying to pick parameters λ to maximize the Lagrangian. Think of the adversary as an auditor that is trying to find the directions in which D is unfair.

   The goal is to compute a Nash Equilibrium of this game. Luckily, there is a well-known, standard approach to computing a Nash to arbitrary precision in zero-sum game. The following dynamics, first studied in [FS96], will converge to an equilibrium:

(a) At each time step t, the adversary/auditor plays a no-regret algorithm to pick a distribution Λ_t over parameter vectors λ's. Here the regret is defined in terms of the Lagrangian loss. One possible no-regret algorithm to use here is Follow the Perturbed Leader.

(b) The learner plays a hypothesis h_t that *best responds* to the adversary's parameters, i.e.

$$D_t = \arg \min_{h \in \mathcal{H}(S)} \mathcal{L}(h, \lambda_t). \tag{1}$$

It is then known that the average distribution $\bar{D}_T$ over time that picks h with probability $\frac{1}{T}\sum_{t=1}^{T} \mathbb{1}\{h_t = h\}$ is an approximate equilibrium strategy for the learner, hence an approximate solution to Fair ERM! The general techniques here are pretty classical, but there are a few subtleties that are beyond the scope of this class. For more details on solving the best response for the learner, on FTPL for the auditor, and on the exact approximate guarantees that are obtained, please check [KNRW18] carefully.

Note that this is a relatively simpler problem than trying to solve the minimization problem directly. [KNRW18] provides more details, but: on the one hand, the best response problem solved by the learner at each time step is a cost-sensitive classification problem; so, the problem can be solve assuming that we have access to a black-box or oracle that solves CSC problems for us. On the other hand, they show how FTLP can be implemented efficiently if we once have access to a CSC oracle. In turn, this provides an *oracle-efficient* (it is computationally efficient, up to the need for an oracle that itself may not be) approach to solving Fair ERM, with accuracy guarantees that are well-understood in terms of how large is T/how long we run our dynamics. Note that it is not unreasonable here to assume that we have a an oracle/black-box that (approximately) solves the problem for us, as there are good heuristics to solve CSC problems.

4. Finally, we need to show that our solution generalizes beyond the in-sample case, for the true distribution of data $\mathcal{P}_n$, both in terms of having a low loss and the fairness constraints being (approximately satisfied). Once again, I will not go into the details here: this is a simple VC-dimension generalization argument, similar to the one we saw in differential privacy (for SmallDB). We know that with finite VC-dimension, empirical losses are guaranteed to uniformly (taking the supremum over the hypothesis class) converge towards expected losses, with an in-sample accuracy of $\sqrt{\text{VC-DIM}/n}$ where n is the number of samples.

2 Multi-Calibration – Tuesday, Nov 16th

This section is based on [HJKRR18].

Here, the goal is to prevent fairness gerrymandering with respect to calibration. We have seen in previous lecture how calibration is especially susceptible to gerrymandering.

Calibrated scores can vastly depend on how we put agents together into a single sub-group with the same (or similar score). For example, it is possible to give a bad, yet calibrated score to an agent with a very low probability of, say, recidivism, by simply grouping this agent with others that are at high risk of recidivism. Another example could be the following: we have two groups, S and T, and agents in S have a small probability (compared to agents in T) to have a high qualification level for a job. One way to obtain calibrated scores is to give all agents in group S the same low score (equal to the probability someone in this group is qualified), and give qualified agents in group T a high score versus unqualified agents in T a low score. This would discriminate against the *qualified* agents in group S.

Multi-calibration will aim to address these issues, in a similar manner than the fairness gerrymandering paper did [KNRW18]: it aims to provide calibration towards every *sub-group* we specify. As before, here, the sub-groups may be overlapping, or a combination of several protected attributes, as before: so, for example, instead of asking scores to be calibrated at the level of a group defined by a single sensitive attribute such as male vs female, or black and white, we can also specify that our scores should be calibrated with respect to sub-groups such as white females, or black men.

2.1 Model and objectives

In [HJKRR18], the authors consider a collection $\mathcal{C}$ of subsets of individuals. Each subset $C \in \mathcal{C}$ corresponds to a subset of individuals for which calibration needs to hold (we will see formally what this means later on). Note that the same individual may be in several groups $C \in \mathcal{C}$. Here, to define the group, we will, similarly to [KNRW18], introduce a function $c_S : \mathcal{X} \to \{0, 1\}$; the function defines group S, in that an individual i in population $\mathcal{X}$ of size N is in group S if and only if $c_S(i) = 1$.

Multi-calibrated predictions For each individual i, we denote $o_i \in \{0, 1\}$ the realized outcome for agent i (whether the agent recidivates, for example). We let $p_i^* = \Pr[o_i = 1]$. The goal is to come up with a prediction $x_i \in [0, 1]$ for the value of p_i^* for each individual i. We write x the vector of all prediction x_i's and p^* the vector of true probabilities p_i^*'s.

We want our predictions to be *multi-calibrated* over $\mathcal{C}$; to define multi-calibration, we first define an approximate version of calibration. For the sake of developing learning algorithms, we may not want to work with a continuous set of values $v \in [0, 1]$; rather, we may want to discretize the space of such scores. We can look at a λ-discretization of the form $\Lambda = \{\lambda/2, 3\lambda/2, \ldots, 1 - \lambda/2\}$, where $\lambda > 0$ is a small grid parameter. We now assign a score $v \in \Lambda$ to each agent, which is a proxy for all the continuous scores we could have considered in $\lambda(v) = [v - \lambda/2; v + \lambda/2]$. In this case, we can define the following definition of (α, λ)-multicalibration:

Definition 4 ((α, λ)-calibration)**.** *Take $v \in [0, 1]$, $S \subset \mathcal{X}$, and a predictor x, and define $S_v(x) = \{i : x_i \in \lambda(v)\}$ the set of all agents in S to which we assign a score of $v \in \Lambda$ (the discretized score space). The prediction vector x is $(\alpha'\lambda)$-calibrated with respect to group S*

if for all v in Λ, and all categories $S_v(x)$ such that $|S_v(x)| \geq \alpha\lambda|S|$, we have that

$$\frac{1}{|S_v(x)|} \cdot \left| \sum_{i \in S_v(x)} x_i - p_i^* \right| \leq \alpha.$$

Note that there is a small departure compared to the definition of exact calibration we saw previously, which would have required that

$$\frac{1}{|S_v(x)|} \cdot \left| \sum_{i \in S_v(x)} x_i - p_i^* \right| = 0 \;\Leftrightarrow\; v = \frac{1}{|S_v(x)|} \cdot \sum_{i \in S_v(x)} p_i^*.$$

I.e., among the agents in group S to which we assign a score of $x_i = v$ (so the agents in S_v), the score is exactly equal to the average probability p_i^* $\frac{1}{|S_v(x)|} \cdot \sum_{i \in S_v(x)} x_i$ across these individuals i with score v. Now, however:

1. We allow calibration to be approximate, in the sense that the score v can just be an approximation of this quantity. This is encoded in the fact that we want the expected value to be less than α, rather than 0.

2. We note that we only look at the $S_v(x)$'s that are of big enough size (i.e. at least a constant fraction of S). We will see later that this is because we will use a potential argument to prove convergence, and that the number of steps we will need for convergence will scale in $N/|S_v|$ (intuitively, the smaller and the more groups we have, the harder the problem is, and the more updates we need). By guaranteeing that $|S_v| = \Omega(|S|) = \Omega(\gamma N)$, we are guaranteeing that our number of updates does not grow with N, the size of the population.

   The specific fraction we use $\alpha\lambda$ is chosen to make this a minor requirement: there can only be (given the discretization) at most $1/\lambda$ categories S_v within S for which $|S_v| \leq \alpha\lambda|S|$. But then there are at most $\frac{\alpha\lambda|S|}{\lambda|S|} = \alpha|S|$ elements for which we do not have calibration. So a $1 - \alpha$ fraction of S is calibrated, which is a reasonable approximation.

We are now ready to define multi-calibration, which simply requires calibration for all defined (possibly overlapping, or defined by some complex combinations of functions of several sensitive attributes) subgroups in $\mathcal{C}$:

Definition 5 ((α, λ)-multicalibration)**.** *Let $\mathcal{C} \subseteq 2^{\mathcal{X}}$ be a collection of subsets of $\mathcal{X}$. A predictor is (α, λ)-multicalibrated on $\mathcal{C}$ if for all $S \in \mathcal{C}$, it is (α, λ)-calibrated with respect to S.*

2.2 Learning multi-calibrated predictors from i.i.d samples

Here, we assume that we have *i.i.d.* samples from the data distribution. I.e., we have the ability to sample an agent i uniformly at random from $\mathcal{X}$. For each agent i we sample, we

observed a *realized*, binary outcome o_i (did the agent commit a crime? Repay his loan?), drawn from the Bernoulli distribution with parameter p_i^*: i.e., we have that $o_i = 1$ with probability p_i^*, the true probability of "good" outcomes.

Guess-and-check oracle We will use these samples to implement a "guess-and-check" oracle, which will be the core of the main algorithm in [HJKRR18]. The goal of the oracle is to decide whether our current score estimate v on group S_v is in fact close to $p_{S_v} = \sum_{i \in S_v} p_i^*$, i.e. to tell us whether we are doing well in terms of calibration for this specific sub-group S and score v.

Definition 6 (Guess-and-check oracle)**.** *Let $\tilde{q} : \ 2^{\mathcal{X}} \times [0,1] \times [0,1] \to [0,1] \cup \perp$. A guess-and-check oracle $\tilde{q}$ with window w_0 satisfies the following condition for all $w \geq w_0$, $S \in \mathcal{X}$, and $v \in [0,1]$:*

- *If $|p_S - |S|v| < 2wN$, then $\tilde{q}(S, v, w) = \perp$.*
- *If $|p_S - |S|v| > 4wN$, then $\tilde{q}(S, v, w) \in [0,1]$.*
- *If $\tilde{q}(S, v, w) \neq \perp$, then $||S| \cdot \tilde{q}(S, v, w) - p_S| \leq wN$.*

So, if the current score is accurate, we simply output $\perp$ to say that we are roughly calibrated. If the current score is inaccurate enough, we output an accurate score $\tilde{q}(S, w)$ instead. It is also possible to be in-between (we are neither too accurate or too inaccurate), in which case there is no requirement: it is fine to output any of the two previous options (either $\perp$ or an accurate score). We will see later in this lecture how to implement such an oracle from samples, at a high level.

Algorithm for multi-calibration via guess-and-check oracles Let $\lambda(v) = [v - \lambda/2, v + \lambda/2)$ the continuous range of scores associated with the discrete values $v \in \Lambda$. The algorithm will work as follows:

- Initialize the scores assigned to the agents: let $x = (1/2, \ldots, 1/2) \in [0,1]^N$
- Repeat
  - For each $S \in \mathcal{C}$, $v \in \Lambda$, $S_v = \{i \in S : \ x_i \in \lambda(v)\}$ with $|S_v| = \beta N \geq \alpha\lambda|S|$, do:
    - Set $\bar{v} = \frac{1}{|S_v|} \sum_{i \in S_v} x_i$.
    - Let $r = \tilde{q}(S_v, \bar{v}, \alpha\beta/4)$.
    - if $r \neq \perp$, update $x_i := x_i + (r - \bar{v})$ for all $i \in S_v$ (project onto $[0,1]$ if necessary).
  - If no S_v is updated, HALT.
- For $v \in \Lambda$, let:
  - $\bar{v} = \sum_{i \in \lambda(v)} x_i$

– For $i \in \lambda(v)$: $x_i := \bar{v}$.

So, the general idea of the algorithm is to start with uniform scores and update them over time until we are calibrated. For each v, we define S_v as a function of all agents that have a score (close to, remember we discretized the space) v. We then compare this score (close to v) to the true average probability p_{S_v} for that group (though our oracle $\tilde{q}$). If the oracle outputs $\perp$, we are already (approximately) calibrated for S_v and everything is fine; if not, we update each x_i by the amount by which we are (roughly) un-calibrated; the window size controls how close we are to being calibrated after such an update. We keep doing so until all S_v's are calibrated; if we halt, calibration is easy to show, but the hard part is to show that we halt in finite/polynomial time.

We can show the following lemma:

Theorem 7. *Let γ be such that for all $S \in \mathcal{C}$, we have $|S| \geq \gamma N$. The above algorithm halts after receiving at most $O(1/\alpha^3\lambda\gamma)$ guess-and-check responses where $r \in [0,1]$, i.e. after doing at most that many updates. When we halt, we obtain a (α, λ)-multicalibrated predictor.*

Here, note that γ is a sliding scale: basically, the smaller the groups can be, the more updates we may need to do. Indeed, one can see γ as a measure of complexity of the problem: with $\gamma = 1$, we are just in the standard calibration setting, as we only have to care about a single group that contains the whole population. When γ becomes smaller, the number of sub-groups possibly considered increases, and also the number of ways for the sub-groups to overlap. When $\gamma = 1/N$, we are effectively using a definition of multicalibration that allows calibration with respect to all possible sub-groups of $\mathcal{X}$ in the worst-case.

A second note is that the fact that we have a finite number of updates also means we must have a finite total number of rounds to run: indeed, we can do at most $O(|\mathcal{C}|/\alpha^3\lambda^2\gamma)$ updates. This means there are at least $O(|\mathcal{C}|/\alpha^3\lambda^2\gamma)$ rounds of the algorithm in which we do an update. Further, if we don't do an update in a round (no S_v is updated), we have to halt. Hence there can only be $O(|\mathcal{C}|/\alpha^3\lambda^2\gamma)$ rounds before we stop. Now, because each round calls the query exactly once for each S and each v, there are $|\mathcal{C}|/\lambda$ query calls in a round. Hence, we run total at most $O(|\mathcal{C}|^2/\alpha^3\lambda^3\gamma)$ queries.

Proof. First, when we halt, it must be that for all S and v, we have $\tilde{q}(S_v, \hat{v}, \alpha\beta/4)) = \perp$. By the definition of our oracles, this means that

$$|p_{S_v} - |S_v|\bar{v}| \leq 4wN = 4\alpha\beta/4 \cdot N = \alpha\beta N = \alpha|S_v(x)|.$$

This can be rewritten

$$\left| \sum_{i \in S_v} p_i^* - x_i \right| \leq \alpha|S_v(x)|.$$

So we are indeed (α, λ)-calibrated.

Now, we want to make sure the algorithm converges (there is a finite number of updates before we halt). The proof goes through a potential argument. We will aim to understand

how the potential function $\|p^* - x\|$ evolves over time. Suppose the current score vector is x, and we update x_i to $x'_i = x_i + r - \hat{v}$ when $r \neq \perp$. We then have, letting $\delta_v = r - \hat{v}$

$$\begin{aligned}\|x - p^*\|^2 - \|p^* - x'\|^2 &= \sum_{i \in S_v} \left((p^*_i - x_i)^2 - (p^*_i - x_i + \delta_v)^2\right) \\ &= \sum_{i \in S_v} \left(2(p^*_i - x_i)\delta_v - \delta_v^2\right) \\ &= -|S_v|\delta_v^2 + 2\delta_v \sum_{i \in S_v} (p^*_i - x_i).\end{aligned}$$

Now, let $\mu = \frac{1}{|S_v|}\sum_{i \in S_v}(p^*_i - x_i) = \frac{1}{|S_v|}\sum_{i \in S_v} p^*_i - \bar{v}$ for simplicity of notations. Note that since we use a guess-and-check oracle, it must be that whenever r is a real number (not $\perp$), we have that, also plugging in $w = \alpha\beta/4$

$$\left|\sum_{i \in S_v} p^*_i - |S_v| r\right| \leq wN,$$

which implies

$$\left|\frac{1}{|S_v|}\sum_{i \in S_v} p^*_i - r\right| \leq wN/|S_v| \leq w/\beta,$$

(the new equality following from $|S_v| = \beta N$), or

$$|\mu - \delta_v| = \left|\left(\frac{1}{|S_v|}\sum_{i \in S_v} p^*_i - \bar{v}\right) - (r - \bar{v})\right| \leq w/\beta.$$

Hence we can write $\mu = \delta_v + \eta$ where $|\eta| \leq w/\beta$. Plugging this in the formula for the progress between x and x', we get

$$\begin{aligned}\|x - p^*\|^2 - \|p^* - x'\|^2 &\geq 2\delta_v\mu - \delta_v^2|S_v| \\ &\geq 2(\mu - \eta)\mu - (\mu - \eta)^2|S_v| \\ &\geq (\mu^2 + \mu\eta - \eta^2)|S_v|.\end{aligned}$$

This is concave in η, hence this is minimized at an extreme value in $[-w/\beta, w/\beta]$, i.e. when $|\eta| = w/\beta = \alpha/4$ (using $w = \alpha\beta/4$ as our window).

Now, note that whenever r is real rather than $\perp$, we have that $|S_v| = \beta N$ with a window of $w = \alpha\beta/4$, so we must have

$$\begin{aligned}\left|\sum_{i \in S_v} p^*_i - x_i\right| &= \left|\sum_{i \in S_v} p^*_i - \bar{v}\right| \\ &\geq 2wN \\ &= 2(\alpha\beta/4)N \\ &= \alpha|S_v|/2,\end{aligned}$$

so $|\mu| \geq \alpha/2$.

Plugging these two bounds on η and μ into our lower bound, we have that the potential function $\|x - p^*\|$ decreases by an amount of at least

$$\begin{aligned}
\|x - p^*\|^2 - \|p^* - x'\|^2 &\geq (\mu^2 + \mu\eta - \eta^2)|S_v| \\
&\geq (\alpha/4)^2 |S_v| \\
&= \alpha^2 \beta N/16 \\
&\geq \alpha^3 \lambda |S|/16 \\
&\geq \alpha^3 \lambda \gamma N/16.
\end{aligned}$$

Note that this is where we use $|S_v| = \Omega(N)$, to guarantee that the change in potential changes by at least $\Omega(N)$. At the same time, $\|x - p^*\|^2 - \|p^* - x'\|^2$ and $\|z\|_2^2 = \sum_{i \in [N]} z^2 \leq N$ for $z = p_i - x_i \in [-1, 1]$. As the potential function is lower bounded by 0, this means we make at most $O(1/\alpha^3 \lambda \gamma)$ updates until we halt, upper bounding the number of non-$\perp$ responses. $\square$

Implementing the oracle What we want to do here is to make sure that our guess-and-check oracle has good generalization guarantees: it needs to be accurate on the true distribution of data p^* just from the samples we see. This is however not a trivial problem here: we are calling this guess-and-check oracle *adaptively*: the parameters we pick S_v and $\bar{v}$ depend on the previous steps of the algorithm. This makes it harder to get good generalization guarantees (adaptive data analysis is known to lead to over-fitting if trying to answer too many queries).

There is however a known, nice connection between generalization in adaptative data analysis, and... once again, differential privacy! I.e., if we implement our check-and-guess oracle in a carefully chosen, differentially private manner, we will get the generalization guarantees that we want!

So, here, we want to do the following: we may want to run k (the maximum number of rounds of the algorithm, as given by Theorem 7) queries $\tilde{q}$ on a sequence of (S_1, v_1, w_1) to (S_k, v_k, w_k) in an adaptive manner. Among those queries, we really only want to answer at most m queries numerically (m here is the maximum number of updates we will do, once again given by Theorem 7), and answer the rest of the queries with $\perp$; we want to answer the queries for which we are very inaccurate, which can be seen as the value of q being above a certain threshold. We also want to do this in a differentially private manner for generalization. This should immediately remind you of Private Multiplicative Weights! PMW exactly does this: it uses SparseVector to differentially-privately find queries that we answer poorly, and update our hypothesis (here x) differntially privately to answer that query more precisely!

I will skip on the details here, but it can be shown by combining i) the guarantees of PMW that show we answer the queries accurately on our *in-sample* database, and ii) generalization bounds of differential privacy to extend beyond the in-sample case to the true distribution of data (see the references provided in [KNRW18], Section 3.3. in particular), that we can implement an algorithm $\mathcal{A}$ that solves the above problem using a tractable number of samples.

References

[FS96] Yoav Freund and Robert E Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the ninth annual conference on Computational learning theory*, pages 325–332, 1996.

[HJKRR18] Ursula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1939–1948. PMLR, 2018.

[KNRW18] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2564–2572. PMLR, 2018.