Bias Mitigation in Machine Learning (Revised 2025)

Ways Things Go Wrong

- Have much less data on some group (fine if groups all "same")
- Different groups have different distributions
- Our features are less predictive on some group
- Some group inherently less predictable
- Our data is biased in the first place

Fairness in ML

- Typically a property of a model (ML algo output)
- Exceptions: online decision-making, RL, bandit settings
- Multiple types of fairness definitions

Types of Model Fairness

- Group fairness (most common)
- Individual fairness
- Interpolations between the two
- Others (causal, fair representations, ...)

Group Fairness Notions

Start by identifying:

- groups or attributes we wish to "protect" (e.g. race, gender)
- What constitutes harm (e.g. error, false pos/neg)

Choices are subjective & domain-specific

Then seek to equalize rates of harm across groups.

Example:

- domain: consumer lending
- groups: male & female
- harm: false rejection (negs)

Want to find model $h(x)$ s.t.

$$FN(h, male) \approx FN(h, female)$$

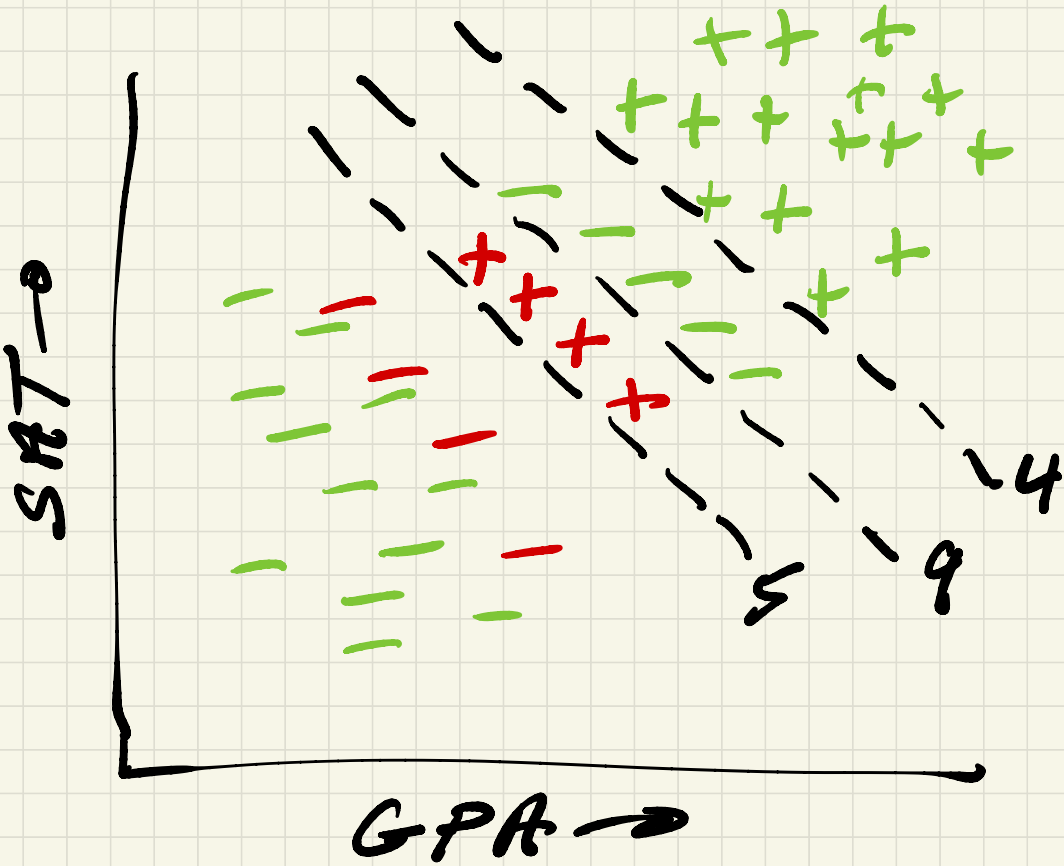
allows for optimization of overall error

If we are given a model $h(x)$ & have access to group membership, easy to audit $h(x)$ for fairness.

How can we learn a fair model $h(x)$?

Why won't standard ML algos work?

A more subtle example



"Bolt-on" Bias Mitigation

- Suppose we are given a model $h \in H$ trained to minimize overall error $\varepsilon = \varepsilon(h)$ within H
- But h is "unfair", meaning that

$$\varepsilon_A = \varepsilon_A(h) = \Pr_{\langle x,y\rangle \sim P}[h(x) \neq y \mid x \in A]$$

$$\varepsilon_B = \varepsilon_B(h) = \Pr_{\langle x,y\rangle \sim P}[h(x) \neq y \mid x \in B]$$

are unequal – say $\varepsilon_A \lneq \varepsilon_B \lneq 1/2$

- Here A & B are two disjoint & exhaustive subgroups, so $A \cap B = \emptyset$, $A \cup B = X$
- Assume $\forall x$ we know whether $x \in A$ or $x \in B$ (might be a feature)
- Goal: Modify h to get a fair model in which error rates on A & B are <u>equal.</u>

Technique: Selective Randomization

- Define a new model $\tilde{h}$ "on top" of h:
  - if $x \in B$, define $\tilde{h}(x) = h(x)$
  - if $x \in A$: for $q \in [0,1]$ TBD, with prob. $1-q$ let $\tilde{h}(x) = h(x)$ with prob. q let $\tilde{h}(x)$ be determined by the flip of a fair coin
- Let $\tilde{\varepsilon}, \tilde{\varepsilon}_A, \tilde{\varepsilon}_B$ denote the error of $\tilde{h}$ overall & on subgroups A & B
- Note $\tilde{\varepsilon}_B = \varepsilon_B$ (didn't change predictions on B)

- Now

$$\tilde{\varepsilon}_A = \underbrace{(1-q)}_{\text{prob. we use } h(x)} \varepsilon_A + \underbrace{q}_{\text{prob. we flip coin}} \underbrace{(1/2)}_{\text{error rate of coin flip}}$$

- Set $\tilde{\varepsilon}_A = \varepsilon_B$ & solve for q:

$$(1-q)\varepsilon_A + q/2 = \varepsilon_B$$

$$q(1/2 - \varepsilon_A) + \varepsilon_A = \varepsilon_B$$

$$\boxed{q = \frac{\varepsilon_B - \varepsilon_A}{1/2 - \varepsilon_A}}$$

At this value of q, we <u>equalize</u> the error rates of $\tilde{h}$ on A & B to be ε_B. The overall error $\tilde{\varepsilon}$ of $\tilde{h}$ will also be ε_B.

Let's do some examples.

- Suppose $\varepsilon_A = 0.05$, $\varepsilon_B = 0.2$. Then

$$q = \frac{0.2 - 0.05}{0.5 - 0.05} = \frac{0.15}{0.45} = 1/3$$

So for group B, we flip a coin $1/3$ of the time

- $\varepsilon_A = 0.1$, $\varepsilon_B = 0.2$:

$$q = \frac{0.2 - 0.1}{1/2 - 0.1} = \frac{0.1}{0.4} = 1/4$$

- So when ε_A & ε_B are closer we randomize less on group B.

- Now suppose we weaken our fairness demand - asking only that

$$\tilde{\varepsilon}_A = \varepsilon_B - \gamma \text{ for some } \gamma > 0$$

- Then we again solve:

$$\tilde{\varepsilon}_A = (1-q)\varepsilon_A + q/2 = \varepsilon_B - \gamma$$

which gives

$$\boxed{q = \frac{\varepsilon_B - \varepsilon_A - \gamma}{1/2 - \varepsilon_A}}$$

Note that q <u>decreases</u> as γ <u>increases</u> - the less fairness we demand, the less we randomize on group B

- Also $\gamma = \varepsilon_B - \varepsilon_A$ is largest sensible value for γ, and corresponds to not changing h at all

- And now the overall error of $\tilde{h}$ will be:

$$\tilde{\varepsilon} = p_A(\varepsilon_B - \gamma) + p_B \varepsilon_B$$
$$= (p_A + p_B)\varepsilon_B - p_A \gamma$$
$$= \varepsilon_B - p_A \gamma$$

Here p_A, p_B are probs. of groups A & B, so $p_A + p_B = 1$

- Example: $\varepsilon_A = 0.05$, $\varepsilon_B = 0.2$, $p_A = 0.7$, $\gamma = 0.03$:

$$q = \frac{\varepsilon_B - \varepsilon_A - \gamma}{1/2 - \varepsilon_A} = \frac{0.2 - 0.05 - 0.03}{1/2 - 0.05}$$

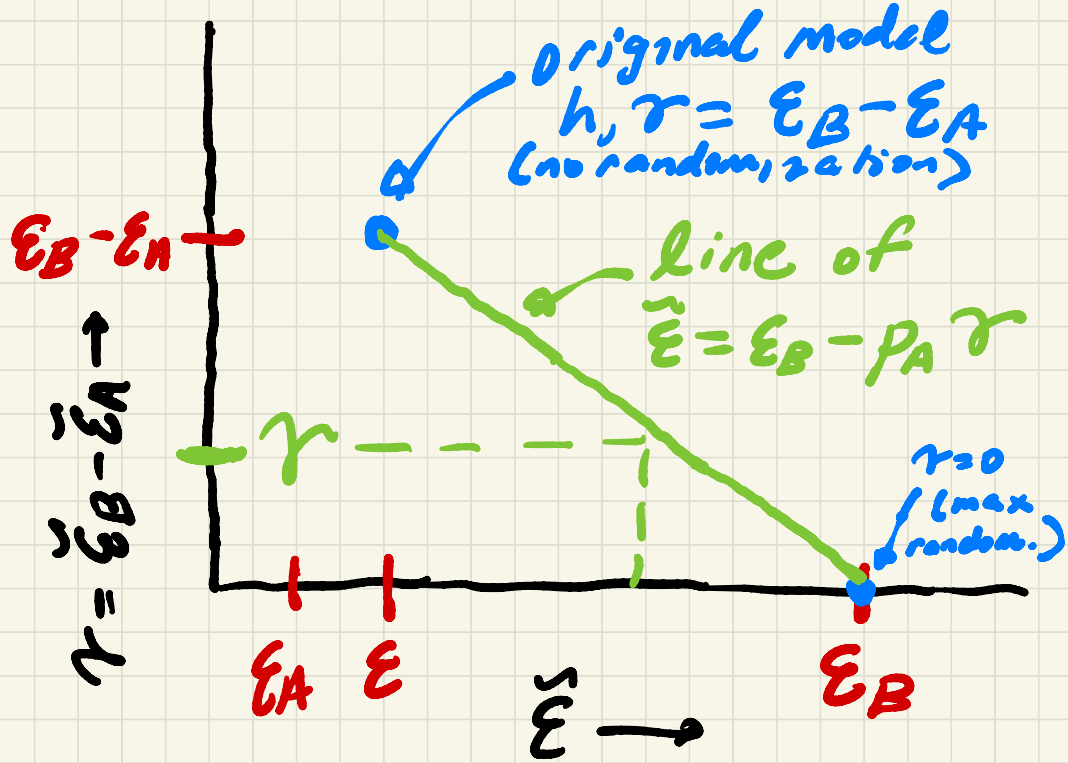
$$= \frac{0.12}{0.45} \approx 0.267$$

and $\tilde{\varepsilon} = \varepsilon_B - p_A \gamma$

$$= 0.2 - 0.7 \times 0.03$$

$$= 0.179$$

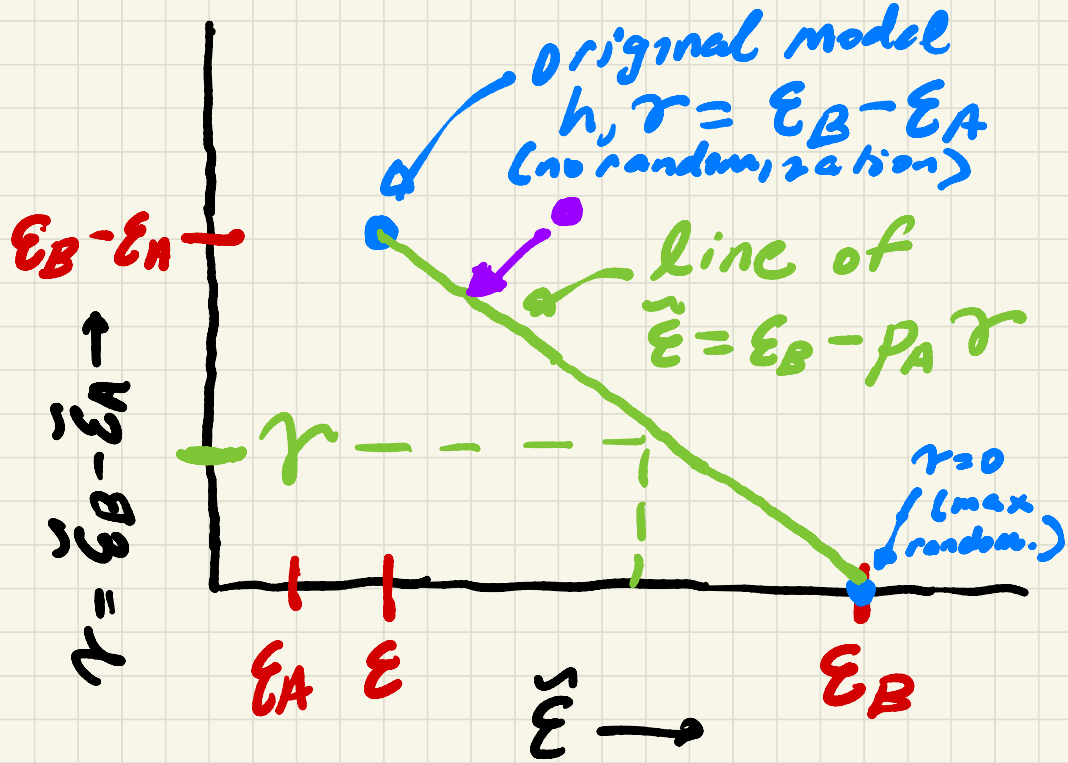
Big Picture: Summary



Slope of line $= \dfrac{\varepsilon_B - \varepsilon_A}{\varepsilon_B - \varepsilon}$

This line gives the achievable trade-offs between error & unfairness in this scheme.

Big Picture: Summary



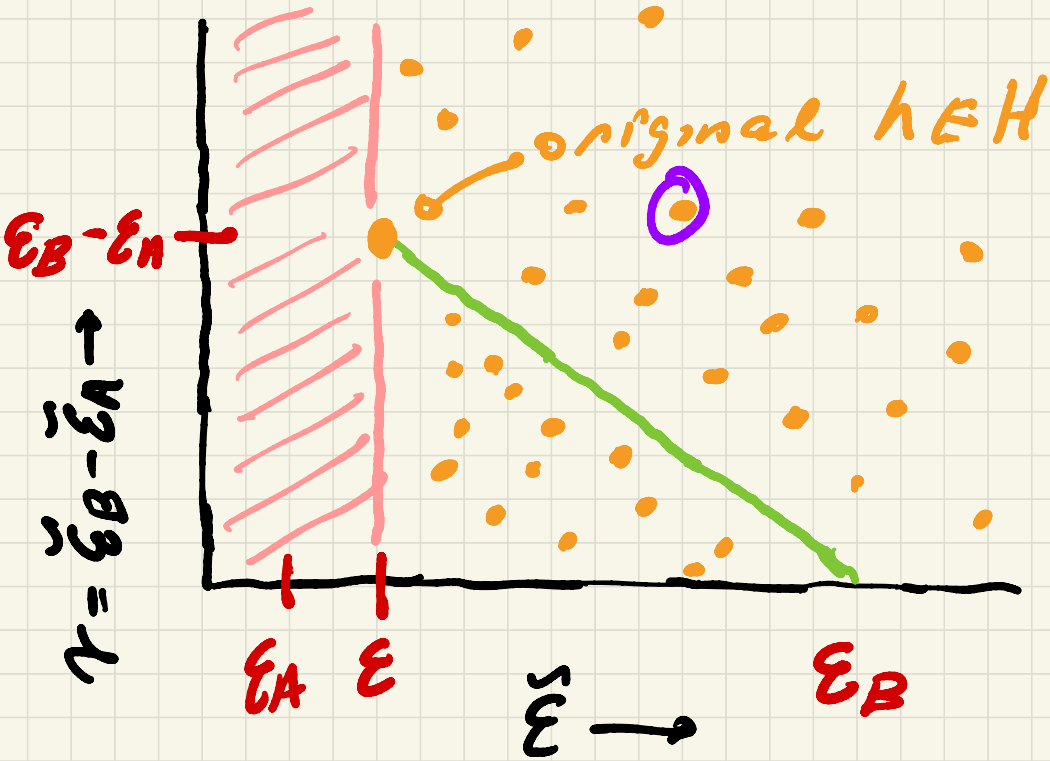
- If we are given a point/model "northeast" of this line, we're better off moving to the line along the perpinduclar-line <u>Pareto</u> <u>dominates</u> the point

Observations

- The bolt-on tradeoff can only improve fairness by deliberately raising the error on the advantaged group
- The bolt-on tradeoff is inherently linear

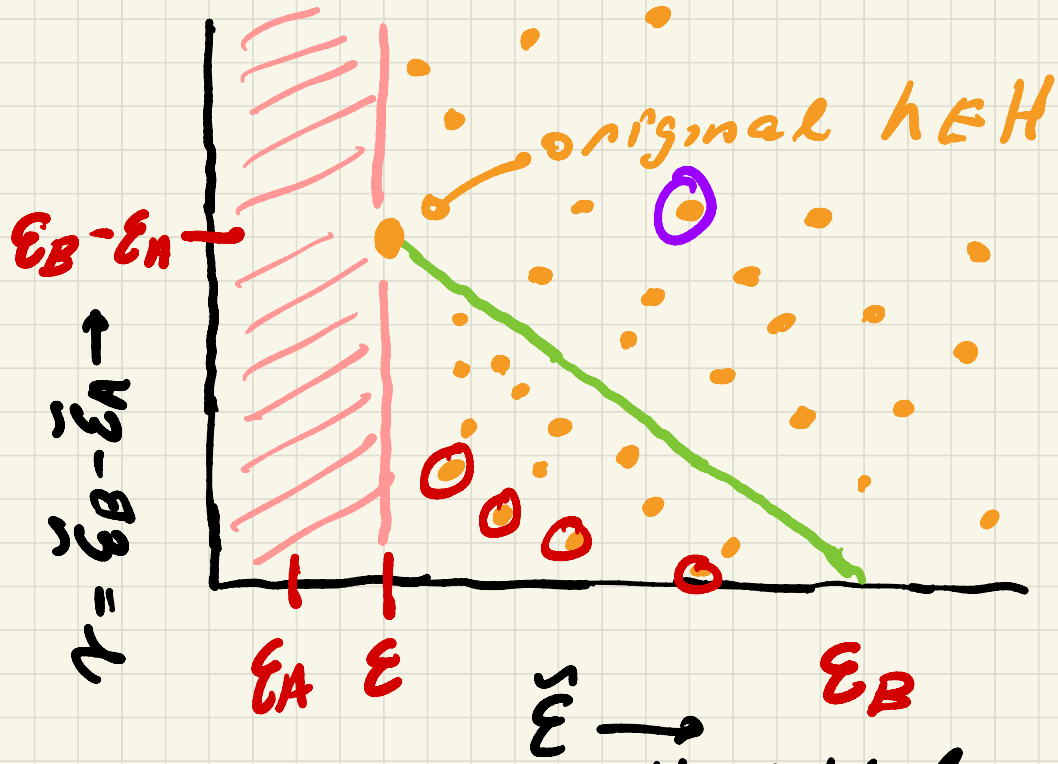
Can we do better?

The H-Frontier



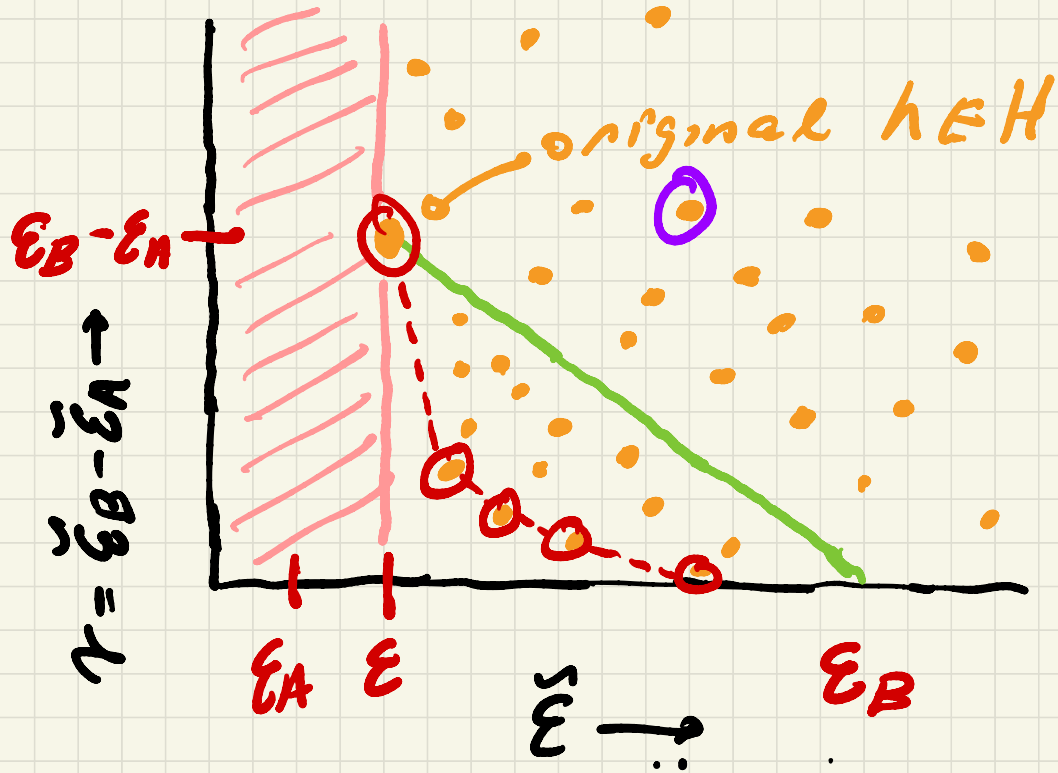
- Imagine we enumerated <u>all</u> the models/points in H
- If original has optimal error in h, no points in shade
- And many points might be Pareto dominated by bolt-on tradeoff

The H-Frontier



- But some points in H might lie <u>below</u> the bolt-on line & also have no points to their SW
- These points have better error & fairness than bolt-on & the other models in H

The H-Frontier



- If we "connect the dots" of these points we have a convex and <u>better</u> tradeoff than bolt-on
- The H-frontier

Connecting the dots:

- Note that if we define

$$g(x) = \begin{cases} h_1(x) & \text{with prob. } \alpha \\ h_2(x) & \text{with prob } 1-\alpha \end{cases}$$

then

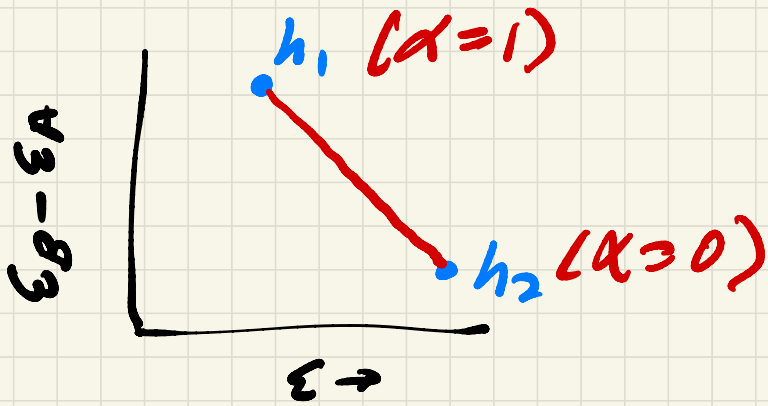
$$\varepsilon(g) = \alpha\,\varepsilon(h_1) + (1-\alpha)\,\varepsilon(h_2)$$

$$\varepsilon_A(g) = \alpha\,\varepsilon_A(h_1) + (1-\alpha)\,\varepsilon_A(h_2)$$

similar for B

$$\Rightarrow \varepsilon_B(g) - \varepsilon_A(g)$$

$$= \alpha\left(\varepsilon_B(h_1) - \varepsilon_A(h_1)\right) + (1-\alpha)\left(\varepsilon_B(h_2) - \varepsilon_A(h_2)\right)$$



How can we find/compute the H-frontier?

Well...

- even finding $h^* \in \mathcal{H}$ is intractable in worst case
- but we do have effective non-fair heuristics

The Reductions/Oracle Approach

- Assume we have a black-box subroutine L for learning $h \in H$ w.r.t. $\varepsilon(h)$ only (non-fair)

- But L is "pretty good" & general (can solve weighted class. problems in H)

Show we can use L for fair learning.

Constrained optimization

$$\min_{h \in \Delta(H)} \{ \varepsilon(h) \} \text{ s.t.}$$

fairness constraints:

(1) $|\varepsilon(h, \text{white}) - \varepsilon(h, \text{black})| \leq \gamma$

(2) $|\varepsilon(h, \text{white}) - \varepsilon(h, \text{hispanic})| \leq \gamma$

(3) $|\varepsilon(h, \text{black}) - \varepsilon(h, \text{hispanic})| \leq \gamma$

$\vdots$

(k) (usually small, but...)

Introduce variables for weights in $\Delta(H)$ & constraints $\Longrightarrow$

huge LP.

- Learner picks $h \in H$
- Regulator picks a pair of groups, e.g. A & B
- Payoff to Regulator:

$$\hat{\varepsilon}(h) + \max(0, |\hat{\varepsilon}(h,A) - \hat{\varepsilon}(h,B)| - \tau)$$

- Payoff to Learner:

$$-(\text{Payoff to Regulator})$$

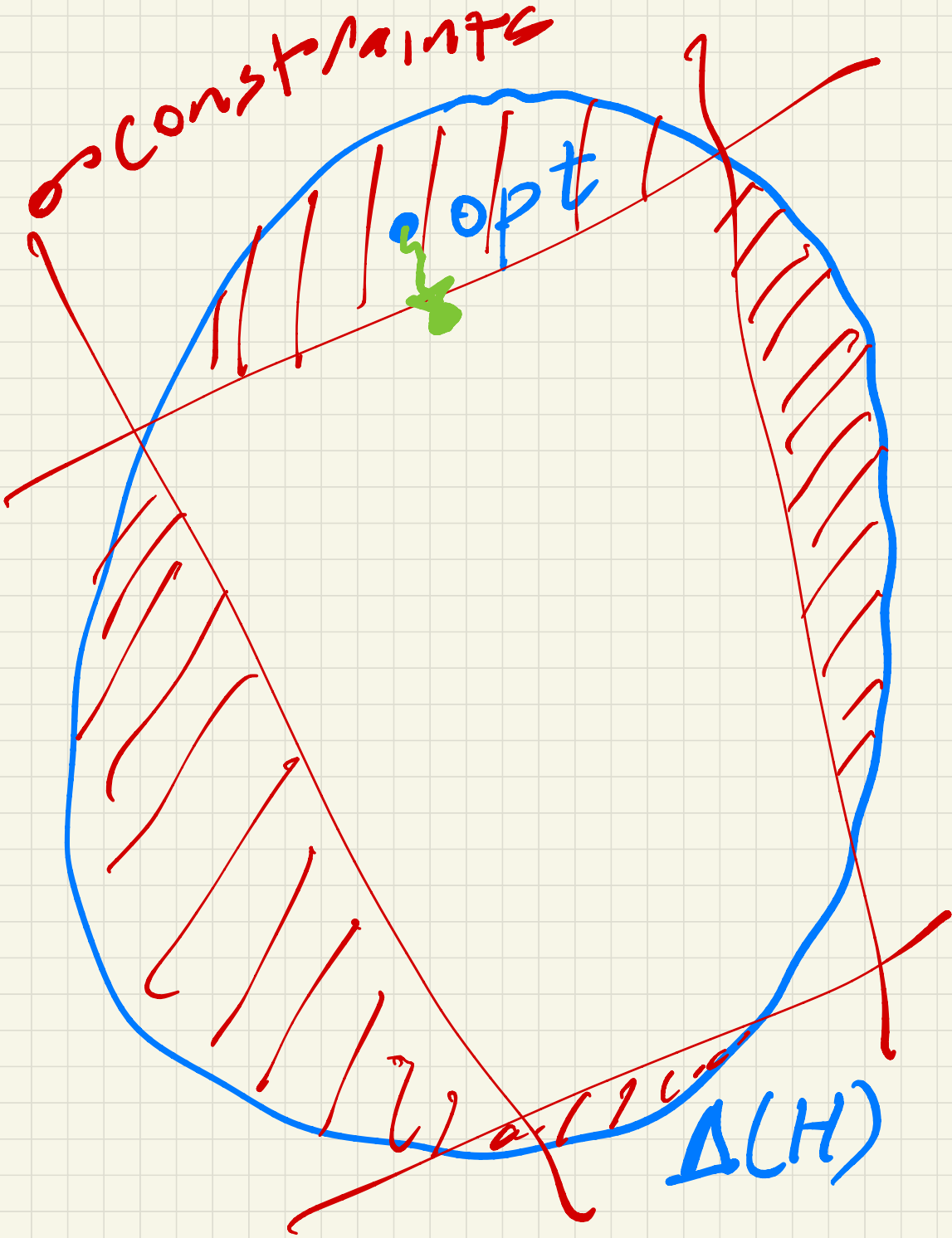
Game Theory Formulation

- Learner plays mixed strategy $p \in \Delta(H)$
- Regulator plays mixed strategy q over fairness constraints
- Zero-sum game on:

$$\underbrace{\varepsilon(p) + \text{constraint violations}(p,q)}_{\text{payoff to Regulator}}$$

$= -$ payoff to Learner

Nash equil = constrained opt solution


constraints
opt
Δ(H)

A Classic Theorem (Freund & Schapire)

If L & R play iteratively:

(1) L best responds to q_t

(2) R updates q_{t+1} using no-regret algo

Then converge to $1/\sqrt{t}$-optimal solution.

(2) usually easy

(1) often reduces to weighted classification with wts. given by g_t

$\Longrightarrow$ "oracle" L.

(Agarwal et al.)

Yields "principled heuristics" that are implementable.

Subgroup Fairness

- Suppose we ask for group fairness by all of race, gender, disability, age, income, ...
- Might still discriminate against disabled Hispanic women over age 55 making $\leq$ 20K/year

Framework

- Model class H
- Group membership class G
- For $g \in G$, $g(x) \in \{0,1\}$ indicates if x is in g (e.g. disabled Hispanic...)
- Now allowing G to be large or infinite

Game Theory II

- Learner plays $h \in H$
- Regulator plays $g \in G$, finds most violated g (e.g. h has high error on g)

Reduce to non-fair case; L no-regret, R best response

Another Approach:

Average Individual Fairness

- Suppose we will make many decisions about x over time
- E.g. product rec's
- Then any h has error rate $\varepsilon_x(h)$ across problems
- Ask that all $\varepsilon_x(h)$ be ≈ equal across individuals x
- Game Theory III

Fairness

Elicitation

- What if fairness isn't "simple"...
- ... but we can elicit empirical fairness judgements.
- E.g.

"Alice & Bob should receive same treatment"

"Alice should be treated at least as well as Bob"

Framework

- Outcome data $S = \{\langle x_i, y_i \rangle\}$
- Fairness data F of form $x_i = x_j$, $x_i \geq x_j$
- Find $h \in H$ that min's error on S subject to F
- Generalize to dist's of S & F
- Game Theory IV

Beyond Equalization

- Problem: may achieve by needlessly inflating harm to advantaged

- Alternative: minimax group fairness:

$$\min_{h \in H} \max_{\substack{\text{groups} \\ g}} \{ \varepsilon_g(h) \}$$

- Game Theory V

$\pi_1, \pi_2 \to$ stat. dist P_1, P_2
& some mix times

- param class $\mathcal{H}$, $\dim(\mathcal{H})$
- $E_{P_1}[(\underline{h_1(x) - v_1(x)})^2] \le \varepsilon$ ✓
  same for P_2
- cross-train:
  $E_{P_1}[(\underline{h_2(x) - v_2(x)})^2] \le \varepsilon$

4 conditions

Alternative Approach:

"Bias Bounties"

Motivation

- AI activism & adversarial dynamics
- Can't defend against everything (?)
- Hard to anticipate groups harmed (gerrymandering)
- "Unexpected" data
- Bug bounties in software

First Idea:

- Invite "crowd" to find problems in a trained model
- E.g. on subsets of training data
- On new data
- "Blurry pictures problem"

Second Idea

- Invite crowd to propose improvements to a trained model
- Submit two classifiers:
  - group classifier $g(x) \in \{0,1\}$
  - model classifier $h(x) \in \{0,1\}$

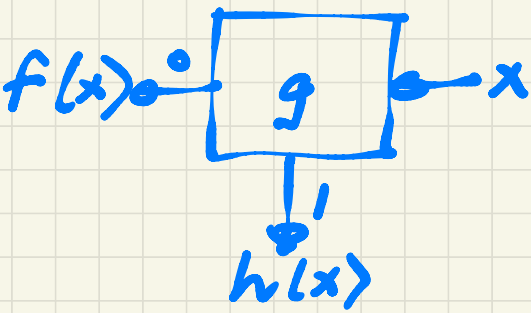
Idea is that $h(x)$ is more accurate on $g(x)=1$ than current model

More formally:

- $f(x)$ is current model
- wrt P, $\varepsilon_g(h) \lneq \varepsilon_g(f)$

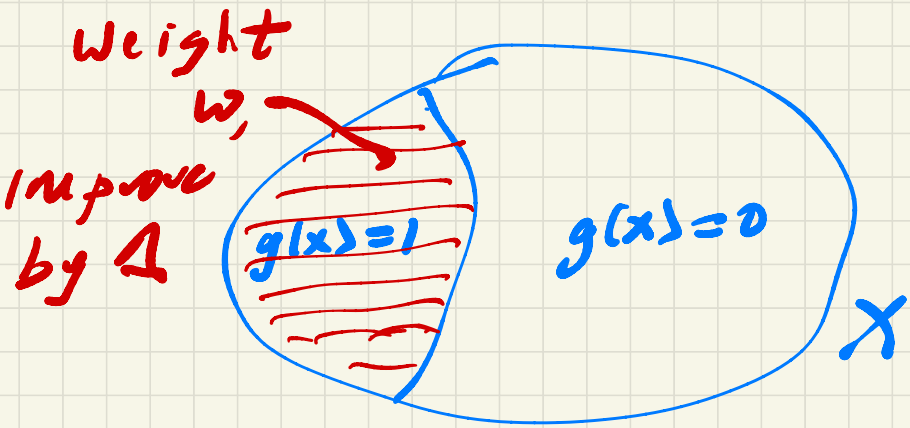
$$\Pr_{x \sim P}[h(x) \neq f^*(x) \mid g(x) = 1]$$

- What to do?

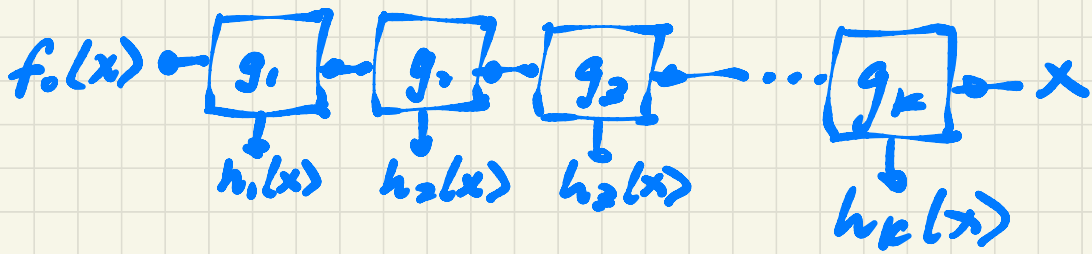


new model
$f_1(x)$

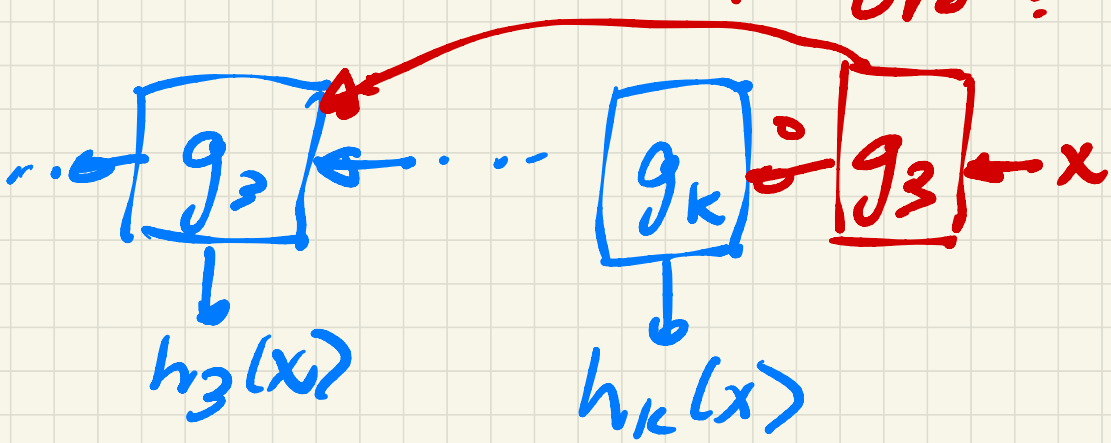
- $\varepsilon_g(f_1) = \varepsilon_g(h) \lneq \varepsilon_g(f)$
- let $\Delta = (\varepsilon_g(f) - \varepsilon_g(f_1)) \gneq 0$
- let $w = \Pr_{x \sim p}[g(x) = 1]$
- $\varepsilon(f_1) = (1-w)\,\varepsilon_{\pi_g}(f)$ no change
  $+ w\,\varepsilon_g(f_1)$ better!
- $\varepsilon(f) - \varepsilon(f_1) = w \cdot \Delta \gneq 0$



Now repeat/recurse:



Hmm... what if g_k makes e.g. g_3 worse off?



"pointer decision list" (PDL)

Ends in one of three ways:

- $\mathcal{E}(f_k) = 0$
- No improving $\langle g, h \rangle$ exists
  - then f_k = Bayes optimal
- Nobody can find improving $\langle g, h \rangle$
  - effective Bayes optimal

Practicalities

- Must give crowd data, not distribution
- What about overfitting?
- One solution:
  - only accept when $w \cdot \Delta \geq \gamma \gneq 0$
  - give no info when (g, h) rejected
  - can only be $1/\gamma$ accepts!
  - use test set to accept/reject

Algorithmic Framework

- Plug in your favorite algo for finding (g, h) improvements
- Ternary classification
- EM-style algo
- Others?

Trade-offs will return...