Understanding Black-box Predictions via Influence Functions

Pang Wei Koh & Perry Liang

Presented by – Theo, Aditya, Patrick

Roadmap

1. Influence functions: definitions and theory

2. Efficiently calculating influence functions

3. Validations

4. Uses cases

Approach

- Reviving an “old technique” from Robust statistics: Influence function
  - Cook & Weisberg (1980): regression models can be strongly influenced by a few cases and reflect unusual features of those cases than the overall relationships between the variables.
    - *Find those influential points.*

- What is the influence of a training point on a model ?
  ”Explain the model through the lens its training data”

Approach

A bit of formalism:

1. $n\ observations, z_i = (x_i, y_i) \in (\mathbb{R}^p,\ \mathbb{R}), i \leq n$
2. $Risk\ function: R: \Theta \to \mathbb{R}: \theta \to \sum L(z_i, \theta)$
3. $Empirical\ risk\ minimizer\ \hat{\theta} = argmin_\theta R(\theta)$
4. Hessian function (applied at $\hat{\theta}$) $H_{\hat{\theta}} = \nabla^2 R(\hat{\theta})$

What do we actually need ?

$f: \mathcal{X} \to \mathbb{R}$ *Let's assume <u>smoothness</u> and <u>regularity</u>*

1. $Taylor - Expansion$: $f(x+h) \underset{h \to 0}{=} f(x) + \nabla_x f \cdot h + o(h)$
2. Chain rule: $\nabla_x f \circ g\ (x) = \nabla_x f(g(x) \cdot \nabla_x g(x)$

Landau notation - $o(h) = \{\epsilon: \mathcal{X} \to \mathbb{R} \mid \frac{\epsilon(h)}{||h||_x} \to 0, ||h|| \to 0\}$

How would the model's prediction change if we did not have this training point ?

- Formally, introduce a perturbation in terms of loss

$$For\ a\ given\ z, \hat{\theta}_{\epsilon,z} = argmin_{\theta}\{R(\theta) + \epsilon L(z,\theta)\}$$

- We are interested in the parameters change when removing z.

$$\begin{cases} \underset{\theta}{\text{argmin}} \sum_{z_i \neq z} L(\mathrm{z}_i, \theta) = \hat{\theta}_{\epsilon=-\frac{1}{n}} \\ \underset{\theta}{\text{argmin}} \sum_{z_i} L(\mathrm{z}_i, \theta) = \hat{\theta}_{\epsilon=0} (= \hat{\theta}) \end{cases}$$

- Change the entire parameters and retrain? Costly.

- Much easier to have a simple approximation

$$\hat{\theta}_h - \hat{\theta}_{\epsilon=0} = \frac{d\theta}{d\epsilon}|_{\epsilon=0} \cdot h + o(h)$$

How would the model's prediction change if we did not have this training point ?

- Need access to: $\frac{d\,\theta_{\epsilon,z}}{d\epsilon}|_{\epsilon=0} \overset{\text{def}}{=} I_{up,params}(z) \overset{notation}{=} (*)$

- MAGIC:

$$\frac{d\theta}{d\epsilon}|_{\epsilon=0} \overset{\text{def}}{=} I_{up,params}(z) = \cdots = -H \cdot \nabla_{\theta} L(z, \hat{\theta})$$

Influence function derivation

- Under the hood: Risk function TWICE-DIFFERENTIABLE, CONVEX

  Hessian: (Diagonalizable) definite positive matrix

- Derivation: 1st-order optimality condition:

$$\nabla_\theta R\left(\hat{\theta}_\epsilon\right) + \epsilon \cdot \nabla_\theta L(z, \hat{\theta}_\epsilon) = 0$$

$$Taylor: \hat{\theta}_0 + \overbrace{(*)\cdot\epsilon}^{} + o(\epsilon) \qquad \hat{\theta}_0 + (*)\cdot\overbrace{\epsilon}^{} + o(\epsilon)$$

$$\nabla_\theta R(\hat{\theta}_0 + (*)\cdot\epsilon + o(\epsilon)) + \epsilon \cdot \nabla_\theta L\left(z, \hat{\theta}_0 + (*)\cdot\epsilon + o(\epsilon)\right) = 0$$

F(x) F'(x).h F(x) F'(x).h

$$\nabla_\theta R(\hat{\theta}_0) + \nabla_\theta^2 R(\hat{\theta}_0)\cdot(*)\cdot\epsilon + \epsilon\{\nabla_\theta L(z, \hat{\theta}_0) + \nabla_\theta^2 L(\hat{\theta}_0)(*)\cdot\epsilon\} + o(\epsilon) = 0$$

$$0 + H\cdot(*)\cdot\epsilon + \nabla_\theta L(z, \hat{\theta}_0)\cdot\epsilon + \nabla_\theta^2 L(z, \hat{\theta}_0)\cdot(*)\cdot\epsilon^2 + o(\epsilon) = 0$$

$$(*) = -(\nabla_\theta^2 L(z, \hat{\theta}_0)\cdot\epsilon + H)^{-1}\cdot\nabla_\theta L(z, \hat{\theta}_0)$$

Linear system+ make $\epsilon \to 0$

Effect on a test point

- Why again? $(*) = \frac{d\theta}{d\epsilon}\Big|_{\epsilon=0} \overset{\text{def}}{=} I_{up,params}(z)$
- Explicit formula: $\hat{\theta}_{-z} - \hat{\theta} = -\frac{1}{n} I_{up,params}(z)$
- How to Compare this change of weights ?

$$I_{up,loss}(z, z_{test}) \overset{\text{def}}{=} \frac{dL(z_{test}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\Big|_{\epsilon=0}$$

$$\text{MAGIC} = \cdots = -\nabla_\theta L(z_{test}, \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})$$

CHAIN RULE

Interpreting the upweight

- Debugging, understanding a model ?
  - What does it mean ? $I_{up,loss}(z, z_{test}) > 0$ $\rightarrow$ Remove z will make the loss higher.

  Geometric interpretation
  - New inner product, a new geometry based on our model's weight

$$-\langle \nabla l(z_{test}) | \nabla l(z) \rangle = -\nabla_\theta L\left(z_{test}, \hat{\theta}\right)^T H_{\hat{\theta}}^{-1} \nabla_\theta L\left(z, \hat{\theta}\right)$$

What do we mean by influence?

- How does it relate to the 'influence' of a point ?
  - Logistic regression model: $p(y|x) = \sigma(y\theta^T x)$

$$I_{up,loss}(z, z_{test})^T = -y_{test} y \cdot \sigma(-y_{test}\theta^T x_{test}) \cdot \sigma(-y\theta^T x) \cdot x_{test}^T H_\theta^{-1} \cdot x$$

  - Influence of high training loss
  - Resistance matrix

Efficiently Calculating the Influence

- Optimally we can determine influence of a training point z_i by leaving out zi, regenerating and assessing the resulting model on z z_{test} (**EXPENSIVE**)
- We came up with **cheaper approximation**

$$I_{up,loss}(z) \; -\nabla_\theta L\left(z_{test}, \hat{\theta}\right)^T H_{\hat{\theta}}^{-1} \nabla_\theta L\left(z, \hat{\theta}\right)$$

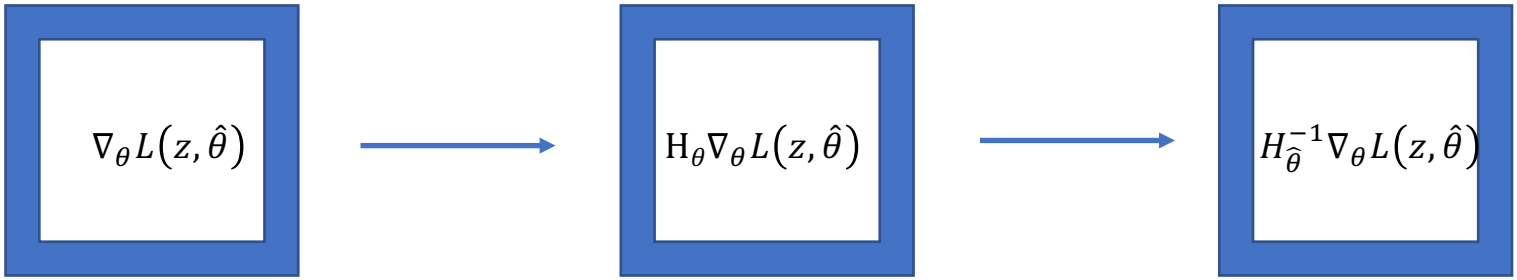
- **Yay??**

Calculating the Inverse Hessian

$$I_{up,loss}(z) \; - \nabla_\theta L\left(z_{test}, \hat{\theta}\right)^T H_{\hat{\theta}}^{-1} \nabla_\theta L\left(z, \hat{\theta}\right)$$

- The **inverse Hessian $\boldsymbol{H_{\hat{\theta}}^{-1}}$** is still quite expensive to compute
- With n training samples and a model with θ in R^p, **$\boldsymbol{H_{\hat{\theta}}^{-1}}$ is $\mathbf{O(np^2 + p^3)}$**
- Remember, we want to find $I_{up,loss}(z)$ (and thus $H_{\hat{\theta}}^{-1}$) for each training point

Inverse Hessian Estimation



Perlmutter ”Trick”

Conjugate Gradients
Stochastic Estimation

Perlmutter 'Trick'

- For Hessian $H_{\hat{\boldsymbol{\theta}}}$ and arbitrary vector $\boldsymbol{v}$ in R^d can calculate $H_{\hat{\boldsymbol{\theta}}}\boldsymbol{v}$ without explicitly knowing $H_{\hat{\boldsymbol{\theta}}}$
- The operation is O(d)

If α is very small and $H_{\hat{\boldsymbol{\theta}}}$ is the Hessian of the function L we can use central difference approximation to formulate $H_{\hat{\boldsymbol{\theta}}}\mathbf{v}$

$$H_{\hat{\boldsymbol{\theta}}}\mathbf{v} \approx \frac{L(x + \alpha\mathbf{v}) - L(x - \alpha\mathbf{v})}{2\alpha}$$

Perlmutter's Trick is also an approximation, but more robust to errors from small α

Conjugate Gradient

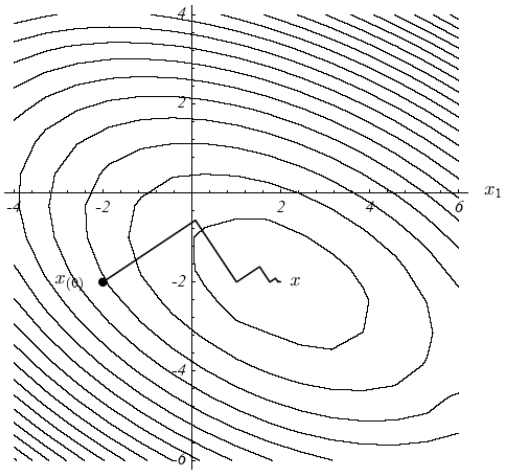
- Now that we know $H_{\hat{\boldsymbol{\theta}}}\boldsymbol{v}$ we want to efficiently construct $H_{\widehat{\theta}}^{-1}\mathbf{v}$
- If we minimize $f(\mathrm{t}) = \frac{1}{2}t^T H_{\widehat{\theta}} t - \mathrm{v}^T$ we'll find $H_{\widehat{\theta}}^{-1}\mathbf{v}$
- At $\mathrm{t}_{\mathrm{min}}$, $0 = \nabla f = H_{\widehat{\theta}} t_{min} - \mathrm{v}$ meaning $t_{min} = H_{\widehat{\theta}}^{-1}\mathrm{v}$

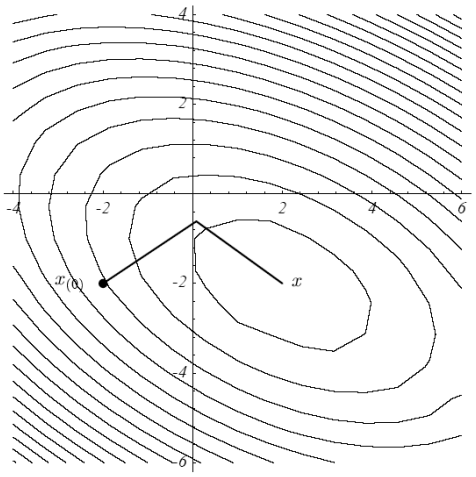
Conjugate Gradient Algorithm

Start with $v_0 \in R^n \; and \; set \; d_0 = r_0 = -\nabla f(v_0)$

- $\alpha_i = \frac{r_i^T r_i}{d_i^T H_{\hat{\theta}} d_i}, r_i = \nabla f(v_i)$
- $v_{i+1} = v_i + \alpha_i \mathrm{d}_i$
- $\beta_i = \frac{r_{i+1}^T r_{i+1}}{r_i^T r_i}$
- $d_{i+1} = r_{i+1} + \beta_{i+1} \mathrm{d}_i$

Repeat n times!!





Problems with CG

- There are problems with CG in particular for models with many parameters
- At each iteration we're doing Hessian evaluations O(p) and we in principle do p iterations.
- As a result, the authors suggest another approximation algorithm for inverting the Hessian -- **Stochastic Evaluation**

Stochastic Estimation

Using a Taylor expansion for $H_j^{-1} \equiv \sum_i (I - H)^i$ and recasting it recursively we have:

$$H_j^{-1} = I + (I - H) H_{j-1}^{-1}$$

This suggests a sampling algorithm to estimate the inverse Hessian based on expectations.

- $Sample\ s\ points\ z_i$
- Use the samples to evaluate H_j^{-1}

Repeat n times!!

Experimental results

ZOOM IN

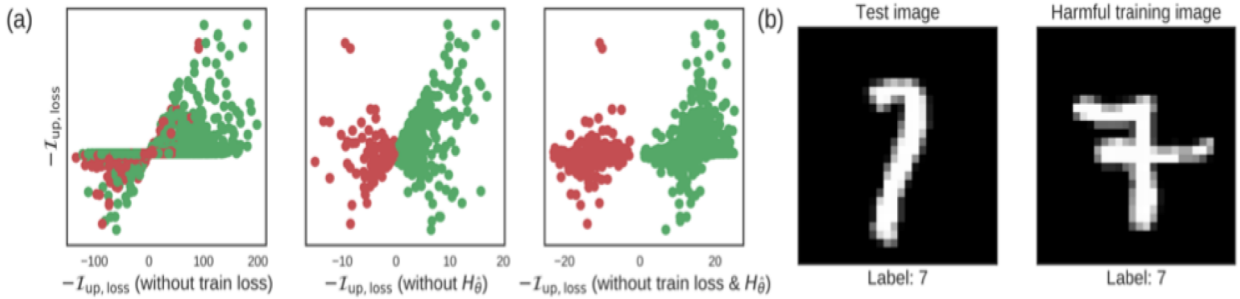
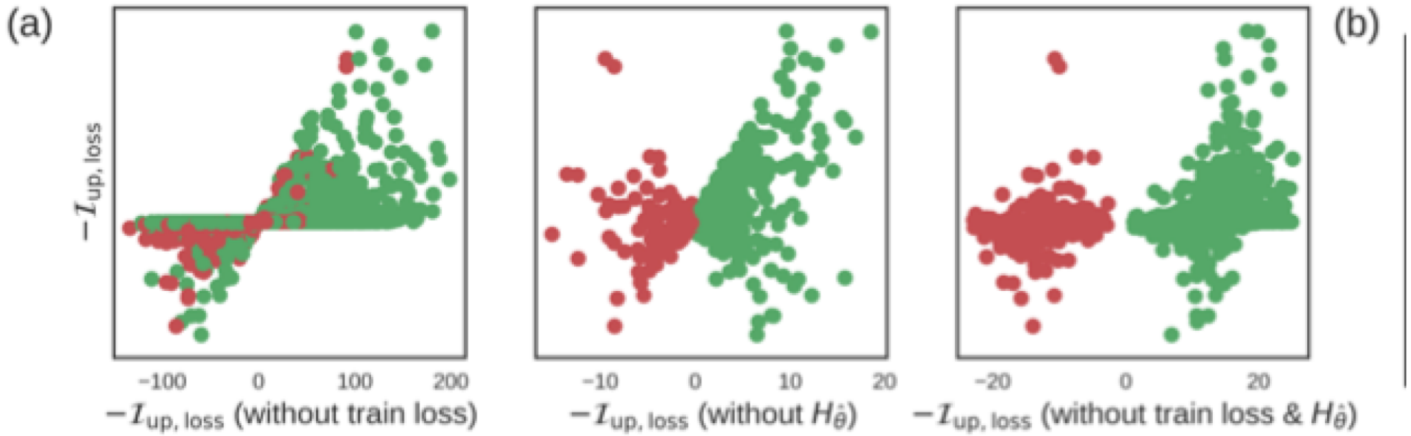


Figure 1. **Components of influence.** **(a)** What is the effect of the training loss and $H_{\hat{\theta}}^{-1}$ terms in $\mathcal{I}_{\text{up,loss}}$? Here, we plot $\mathcal{I}_{\text{up,loss}}$ against variants that are missing these terms and show that they are necessary for picking up the truly influential training points. For these calculations, we use logistic regression to distinguish 1's from 7's in MNIST (LeCun et al., 1998), picking an arbitrary test point z_{test}; similar trends hold across other test points. Green dots are train images of the same label as the test image (7) while red dots are 1's. **Left:** Without the train loss term, we overestimate the influence of many training points: the points near the y=0 line should have $\mathcal{I}_{\text{up,loss}}$ close to 0, but instead have high influence when we remove the train loss term. **Mid:** Without $H_{\hat{\theta}}^{-1}$, all green training points are helpful (removing each point increases test loss) and all red points are harmful (removing each point decreases test loss). This is because $\forall x, x \succeq 0$ (all pixel values are positive), so $x \cdot x_{\text{test}} \geq 0$, but it is incorrect: many harmful training points actually share the same label as z_{test}. See panel (b). **Right:** Without training loss or $H_{\hat{\theta}}^{-1}$, what is left is the scaled Euclidean inner product $y_{\text{test}} y \cdot \sigma(-y_{\text{test}} \theta^\top x_{\text{test}}) \cdot x_{\text{test}}^\top x$, which fails to accurately capture influence; the scatter plot deviates quite far from the diagonal. **(b)** The test image and a harmful training image with the same label. To the model, they look very different, so the presence of the training image makes the model think that the test image is less likely to be a 7. The Euclidean inner product does not pick up on these less intuitive, but important, harmful influences.

- MNIST
- Influence function compared to… Euclidian distance ? $x_{\text{test}}^T \cdot x$

Experimental results



$$(a) - y_{test} y \cdot \sigma(-y_{test}\theta^T x_{test}) \cdot \sigma(-y\theta^T x) \cdot x_{test}^T H_\theta^{-1} \cdot x$$

$$(b) - y_{test} y \cdot \sigma(-y_{test}\theta^T x_{test}) \cdot \sigma(-y\theta^T x) \cdot x_{test}^T H_{\hat{\theta}}^{-1} \cdot x$$

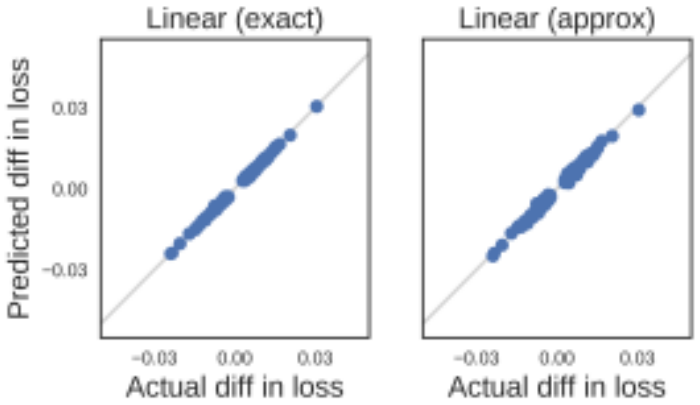
$$(c) - y_{test} y \cdot \sigma(-y_{test}\theta^T x_{test}) \cdot \sigma(-y\theta^T x) \cdot x_{test}^T H_{\hat{\theta}}^{-1} \cdot x$$

Comparison with leave one out (logistic)

Trained basic logistic regression on MNSIT
For a given misclassified z_{test}

Compared every $|I\{z, z_{test}\}||$ for every $z \in z_{train}$
For the top 500 compare $-\frac{1}{n}\ |I_{z,z_{test}}||$ vs change in loss with z removed and retrained.

Tested with both conjugate gradient (left) and stochastic gradient (right)



Comparison with leave one out – non convexity (CNN)

For non-convex example on SGD

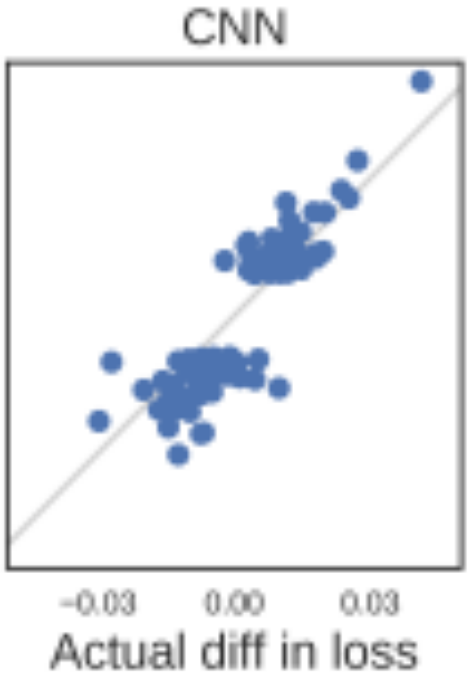
For output of SGD (local not global max) replace Loss function with second order convex approximation

$$L(z,\theta) + \nabla L(z,\widetilde{\theta})^T(\theta - \tilde{\theta}) + \frac{1}{2}(\theta - \widetilde{\theta})^T(H_\theta + \lambda I)(\theta - \hat{\theta})$$

Where $\tilde{\theta}$ is the resulting parameters from SGD and λ is a damping term if H_θ is not Positive definite (convexifying it)

Claim – if $\tilde{\theta}$ is close to the true optimal then this approximation is close to a Newton Step

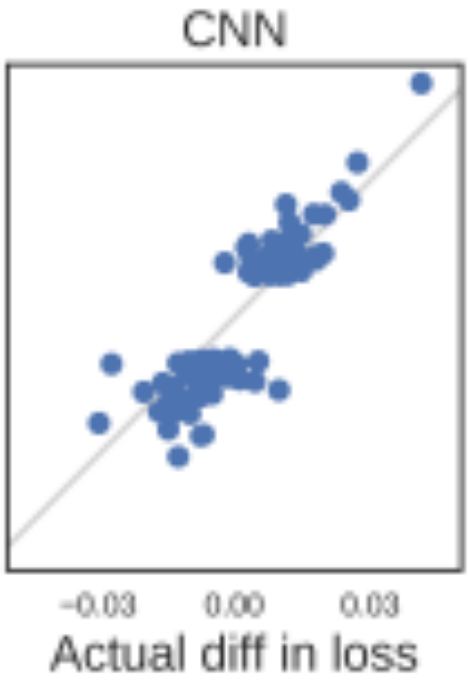
Heavily relies on $\tilde{\theta}$ being close to the true optimal (no clarification on how close)



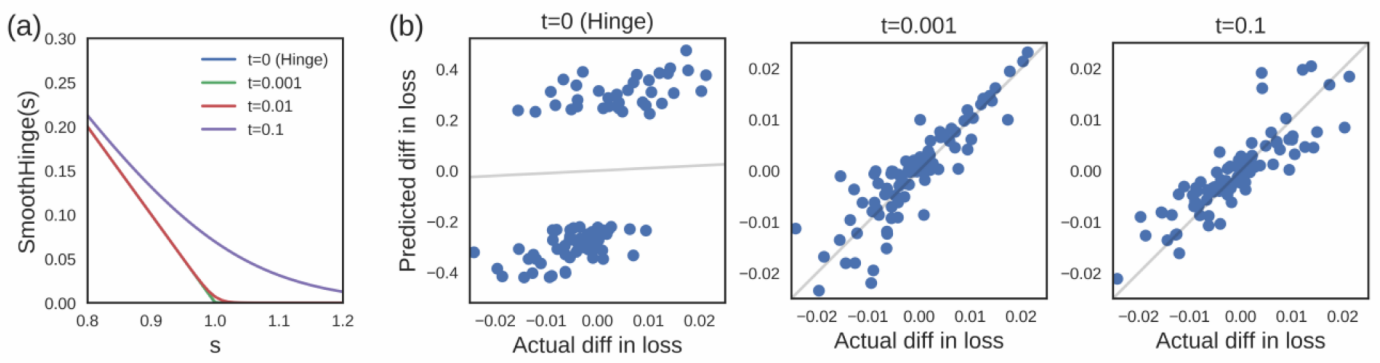
Comparison with leave one out – non convexity (CNN)

Compute $|I\{z, z_{test}\}||$ with new loss function and then see how well it compares to leave one out.
Tested on CNN – compared influence function vs output of includes function. (right)

Pearson Correlation = 0.86 respectively high correlation High correlation



Non-differentiable losses



$$Hinge(s) = \max(0\,, 1 - s)$$

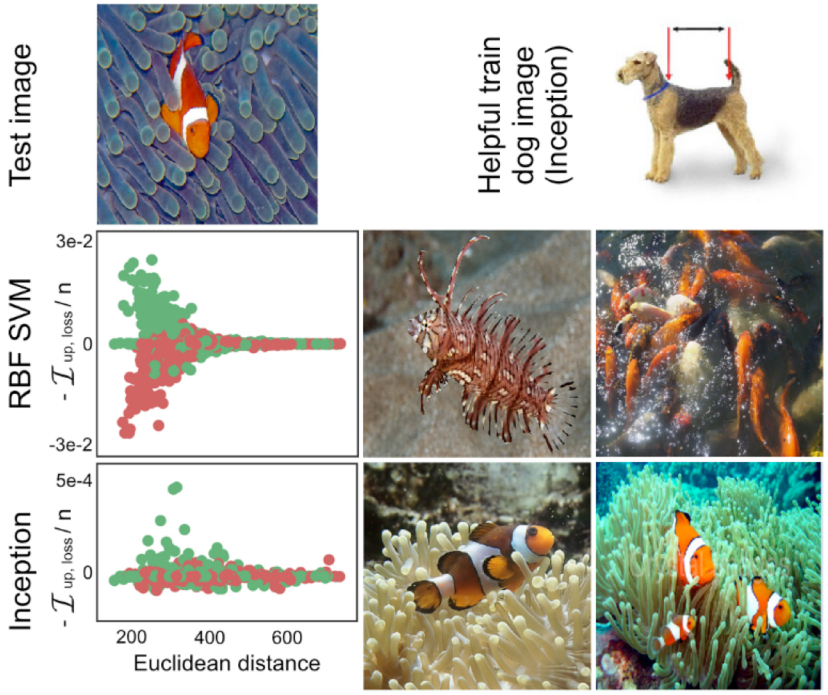
$$SmoothHinge(s,t) = \mathrm{tlog}(1 + \exp(1 + \exp\left(\frac{1-s}{t}\right))$$

- Key idea: Train the initial model on your non-differentiable loss, use a smooth approximation for the influence
- Scalable?

Understanding Model Behaviour

- Task: Classifying Dog vs Fish
- Two Models
  - Logistic Regression Model on top of Inception v1 features
  - RBF SVM Model

Understanding Model Behaviour



How would the model's predictions change if the training input were modified ?

- Formally, introduce the perturbation $z_\delta = (x + \delta, y)$

  No need for δ to be infinitesimal

- New optimal parameters, new risk function

  $\hat{\theta}_{\epsilon,z_\delta,z} = argmin_\theta\{R(\theta) + \epsilon L(z_\delta, \theta) - \epsilon L(z, \theta)\}$

- Under the hood ? Exact same derivation.

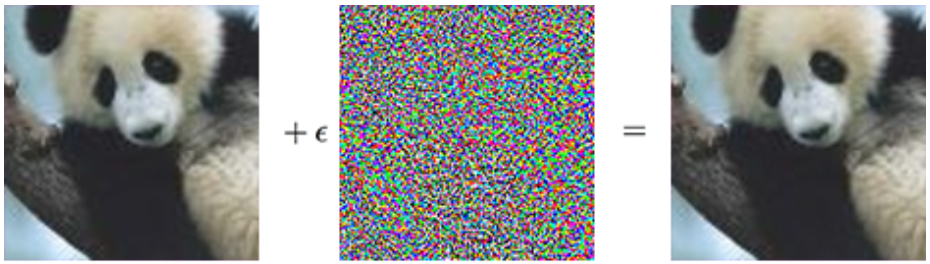
  Explicit formula: $\hat{\theta}_{\epsilon,z_\delta,z} - \hat{\theta} = -\frac{1}{n}(I_{up,params}(z_\delta) - I_{up,params}(z))$

- One final approximation if we make δ infinitesimal:

$$I_{pert,loss}(z, z_{test}) \stackrel{\text{def}}{=} \left.\frac{dL\left(z_{test}, \hat{\theta}_{\epsilon,z_\delta,z}\right)}{d\epsilon}\right|_{\epsilon=0}$$

$$\text{MAGIC} = \cdots = -\nabla_\theta L\left(z_{test}, \hat{\theta}\right)^T H_{\hat{\theta}}^{-1} \nabla_x \nabla_\theta L\left(z, \hat{\theta}\right)^T$$

Detecting robustness to adversarial training attacks



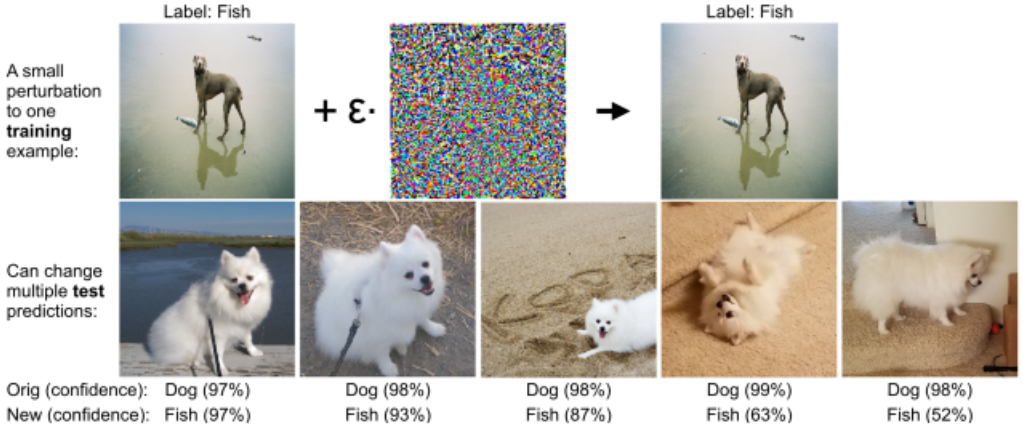
"panda"
57.7% confidence

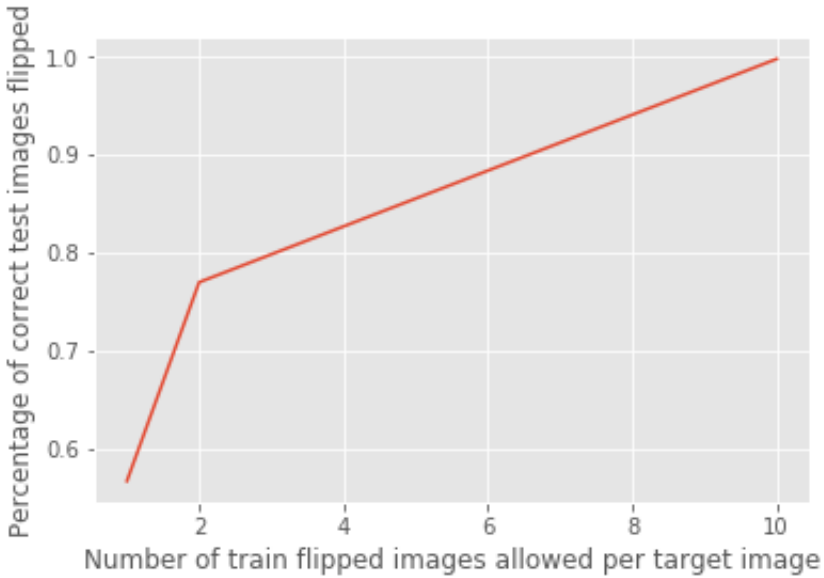
"gibbon"
99.3% confidence

- Standard adversarial attack -> attack the training dataset
- Claim: Use influence method to determine which *training* points are suspectable to adversarial
- Recall : $I_{\{perf,loss\}}(z, z_{test}) = -\nabla_\theta L(z_{test}, \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_x \nabla_\theta L(z, \hat{\theta})^T$
- Interpretation : modify training point z to change the loss of z_{test}
- Claim –equivalent an existing gradient based method but the proposed method will search through perturbations that result in interpretably different images

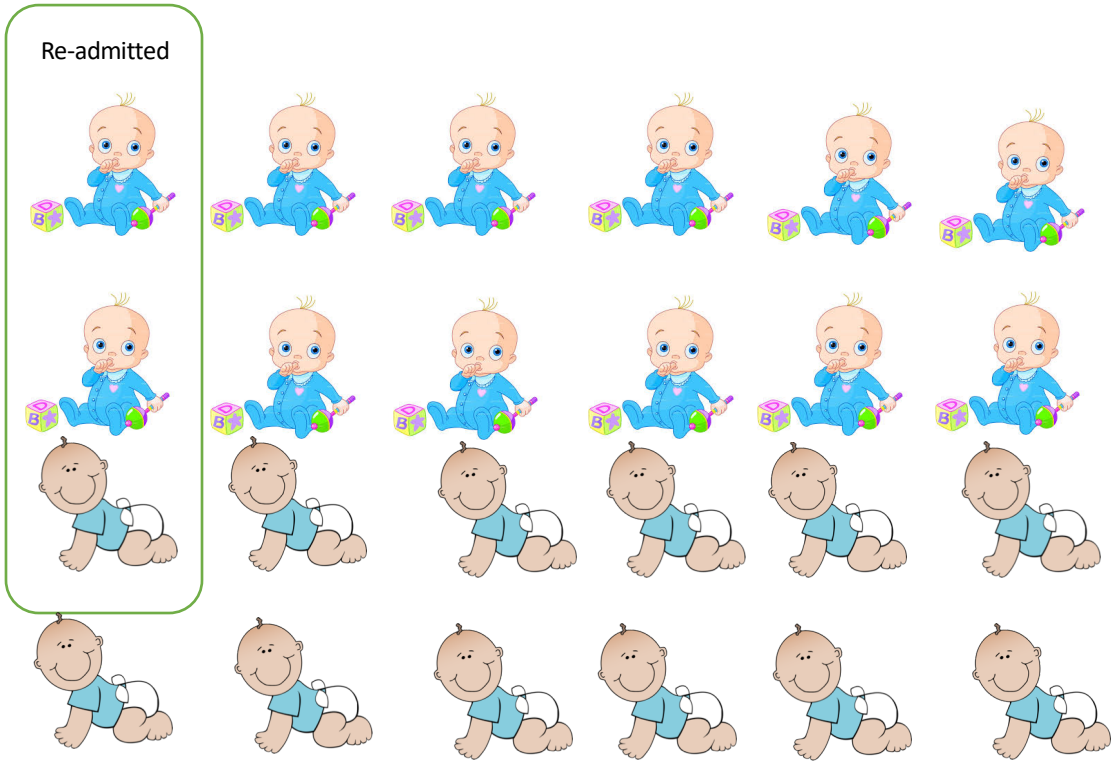
Detecting robustness to adversarial training attacks

- Search for $\widetilde{z_i}$ that is visually indistinguishable of z_i (contains the same 8-bit representation)
- Test case - Inception net.
  - 591/600 accuracy on test without attack
- For correctly classified images search for a training image (z_i) (out 1800 points) where there is a $\widetilde{z_i}$ that is visually the same but flips the prediction
- Ambiguous and mislabeled examples prime vectors of attack



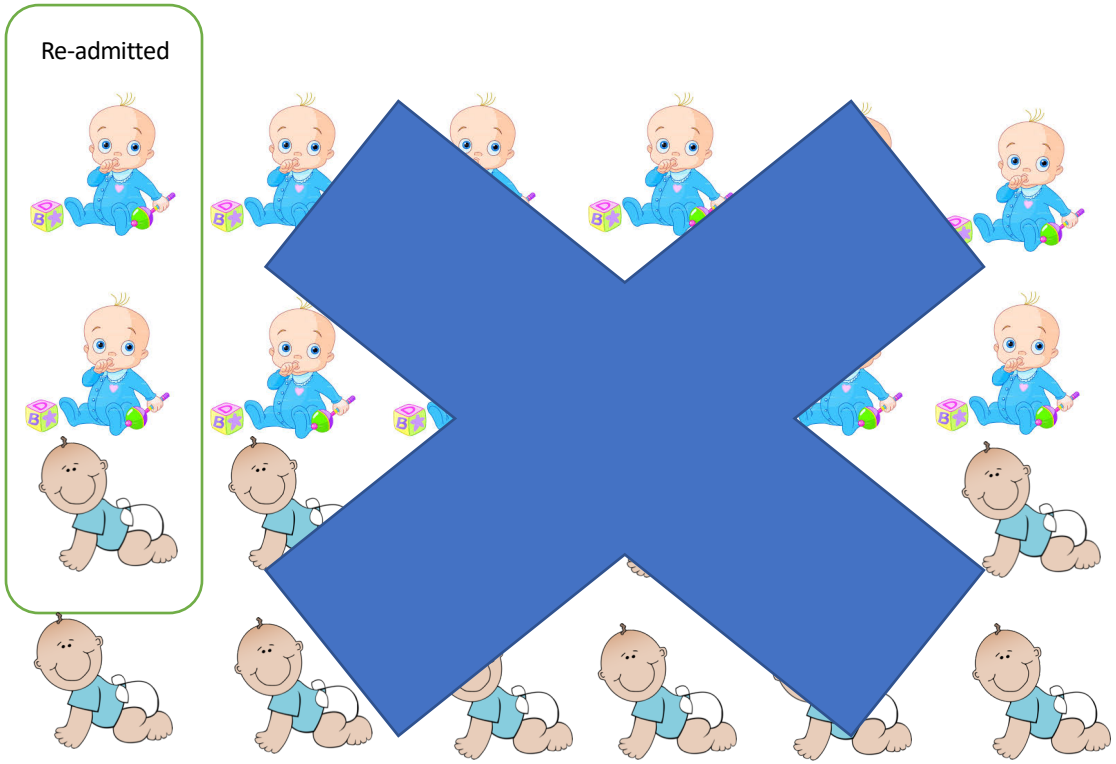


Debugging Domain Mismatch



- 20k dataset across 127 hospitals – use logistic regression to determine readmission
  - "Is child" a binary feature
- 3/24 kids under 10 readmitted

Debugging Domain Mismatch

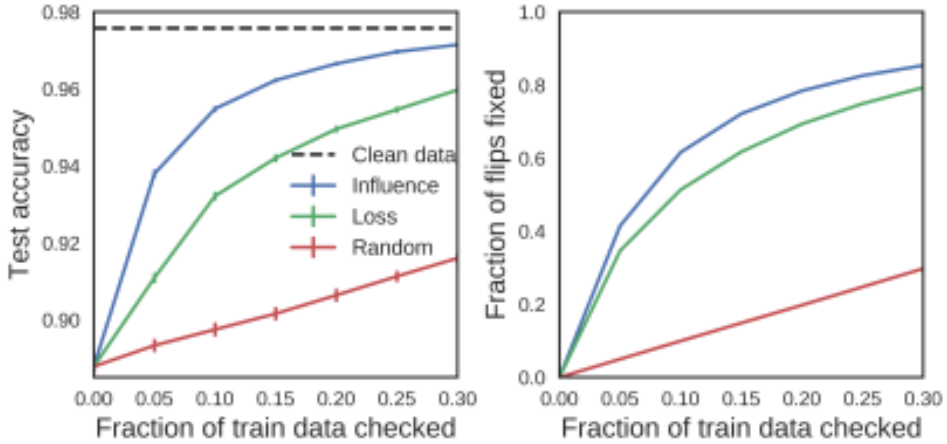


- Modified dataset 20 kids that weren't readmitted taken out of the dataset
- Retrained – got worse accuracy
- Coefficients of logistic regression were less informative (expected "child" to be most important)
- Compute influence function with a random incorrectly labeled test point
- With influence function were able to tell that the 4 children in training were 30-40 times more influential and that the child indicator variable extremely important

Fixing Training Data

Training data labels can be noisy/subject to attacks
Can use influence functions to “diagnose” important points and verify that they’re labeled accurate
Claim – can compute this on just the training set $I(z_i, z_i)\ \forall z_i \in Z_{train}$

Experiment: Flip 10% of labels in a training dataset and sort through the points to flip using various sorting's (random, .loss, influence).



Wrap up

- ***Look at the model as a function of the input data rather than fixed***
- Using influence functions helps describe a model's behavior with respect to its input data
  - Can efficiently compute using stochastic methods
  - Compared with leave one out training more efficient with ways efficient Hessian computation

- Strong assumptions on convexity/differentiability
  - Can attempt convexify/approximate around the problem

- Applications
  - Help identify domain mismatches
  - Verify robustness of model with respect to training data

Simplicity Creates Inequity: Implications for Fairness, Stereotypes, and Interpretability

Jon Kleinberg and Sendhil Mullainathan, 2019

Paper Presentation
Zilin Ma
Hayoun Oh
Jazz Zhao

Motivation & Contribution

Motivation

- **Domain Applications:** algorithms in high-stakes decisions
  - e.g. hiring, admissions, lending, bail
  - decisions based on applicants' features, but certain groups are disadvantaged

- **Behavioral Sciences:** negative stereotypes

- **Computer Science:** interpretability ...
  - ... can help detect unfairness or bias (Doshi-Velez & Kim, 2017)
  - ... may have a trade-off with accuracy/efficiency

How do interpretability, efficiency, and fairness relate to each other?

Main Contributions

- formal model for relationship between simplicity and equity
- framework for producing simple prediction functions (SPFs)
  - simplicity ➜ interpretability
  - equity ➜ fairness

- SPFs are strictly Pareto-dominated wrt equity and efficiency
  - increase complexity ➜ strictly increase both efficiency and equity
- SPFs incentivize use of group membership (if disadvantaged group exists)
  - disadvantage ➜ explicit bias

Main Contributions

- formal model for relationship between simplicity and equity
- framework for producing simple prediction functions (SPFs)
  - simplicity ➜ interpretability
  - equity ➜ fairness

- SPFs are strictly Pareto-dominated
  - increase complexity ➜ strictly incr
- SPFs incentivize use of group mer
  - disadvantage ➜ explicit bias

A policy x ***Pareto dominates*** another policy y if two conditions are satisfied:

1. No one strictly prefers y to x—that is, for all i, $U_i(x) \geq U_i(y)$.
2. At least one person strictly prefers x to y—that is, for at least one i, $U_i(x) > U_i(y)$.

If one policy Pareto dominates another, everyone should be able to agree that policy is better

Ethan Bueno de Mesquita, U Chicago

Main Contributions

- formal model for relationship between simplicity and equity
- framework for producing simple prediction functions (SPFs)
  - simplicity ➜ interpretability
  - equity ➜ fairness

- SPFs are strictly Pareto-dominated wrt equity and efficiency
  - increase complexity ➜ strictly increase both efficiency and equity
- SPFs incentivize use of group membership (if disadvantaged group exists)
  - disadvantage ➜ explicit bias

The Model

Productivity

$$\left(x^{\langle 1 \rangle}, \ldots, x^{\langle k \rangle}, \gamma\right) = (x, \gamma) \longrightarrow f(x, \gamma)$$

boolean features and group membership $\gamma \in \{A, D\}$

productivity

Productivity

$$\left(x^{\langle 1 \rangle}, \ldots, x^{\langle k \rangle}, \gamma\right) = (x, \gamma) \longrightarrow f(x, \gamma)$$

boolean features and group membership $\gamma \in \{A, D\}$

productivity

Objective:
rank applicants based on productivity, admit top r fraction

Productivity

$$\left(x^{\langle 1 \rangle}, \ldots, x^{\langle k \rangle}, \gamma\right) = (x, \gamma) \longrightarrow f(x, \gamma)$$

boolean features and group membership $\gamma \in \{A, D\}$

productivity

$$f(x, A) = f(x, D)$$
$$f(x) \neq f(x') \; \forall x \neq x'$$

require

Objective:
rank applicants based on productivity, admit top r fraction

Genericity Assumption

For two sets of rows S, T

$$(S \neq T \wedge S = \{(x, A)\} \implies T \neq \{(x, D)\}) \implies f|S \neq f|T$$

where $f|S$ is the weighted average of $f(x)\ \forall x \in S$.

Genericity Assumption

For two sets of rows S, T

$$(S \neq T \wedge S = \{(x, A)\} \implies T \neq \{(x, D)\}) \implies f|S \neq f|T$$

where $f|S$ is the weighted average of $f(x)\ \forall x \in S$.

What constraints does it impose on productivity?

Disadvantage Condition

Suppose $f(x) > f(x')$ for some x, x'. Then $\dfrac{\mu(x, A)}{\mu(x, D)} > \dfrac{\mu(x', A)}{\mu(x', D)}$.

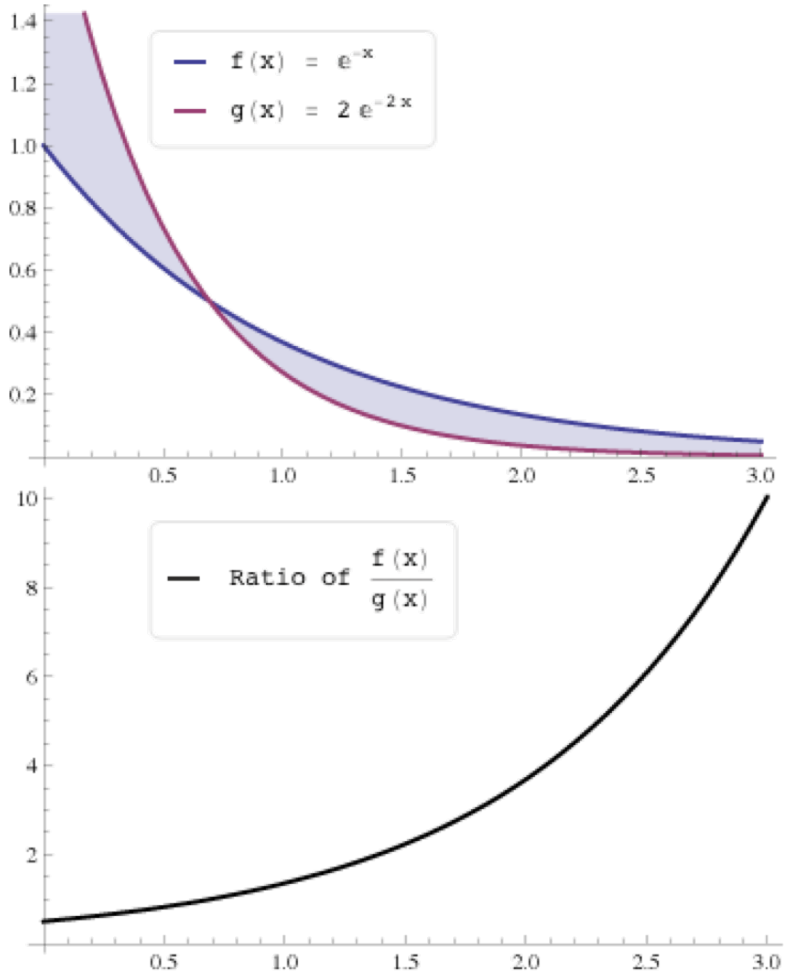
Disadvantage Condition

Suppose $f(x) > f(x')$ for some x, x'. Then $\dfrac{\mu(x, A)}{\mu(x, D)} > \dfrac{\mu(x', A)}{\mu(x', D)}$.

Alternatively: Let $g_A(f) = \dfrac{\mu(x, A)}{\mu(\cdot, A)}, f(x) = f$, same for D.

Then $f > f' \implies \dfrac{g_A(f)}{g_D(f)} > \dfrac{g_A(f')}{g_D(f')}$ or $\dfrac{d}{df}\left(\dfrac{g_A(f)}{g_D(f)}\right) > 0$ (if differentiable)

Disadvantage Condition



Source: https://en.wikipedia.org/wiki/Monotone_likelihood_ratio#/media/File:MLRP-illustration.png

Disadvantage Condition

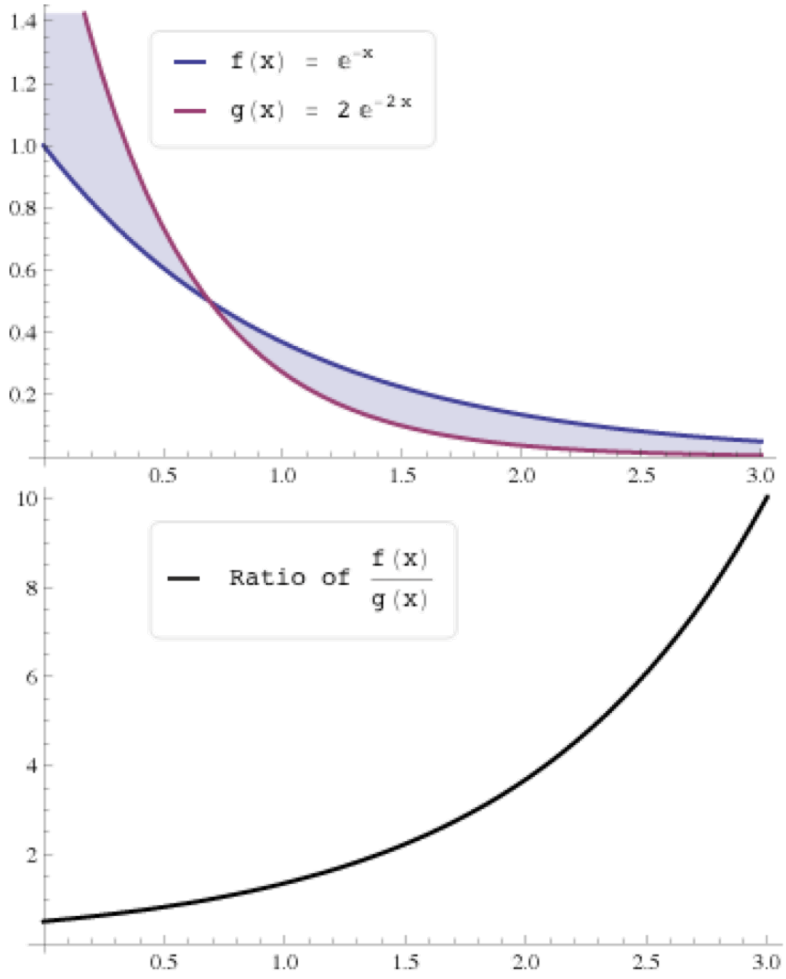
Requiring $f|U_A > f|U_D$ is not enough (Simpson's Paradox)

While $E_x[f(x, A)] > E_x[f(x, D)]$

$\exists x^{\langle j \rangle} s.t.$

$E_{x^{\langle -j \rangle}}[f(x, A)|x^{\langle j \rangle}] < E_{x^{\langle -j \rangle}}[f(x, D)|x^{\langle j \rangle}]$

where $p(x, \gamma) = \frac{\mu(x, \gamma)}{\mu(\cdot, \gamma)}$



Source: https://en.wikipedia.org/wiki/Monotone_likelihood_ratio#/media/File:MLRP-illustration.png

Approximators

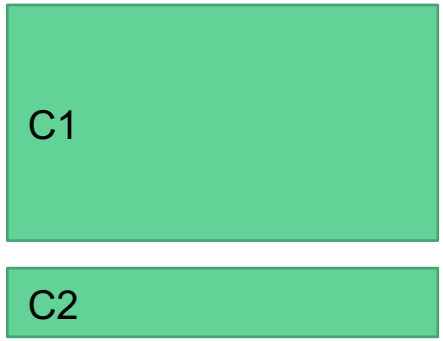
Well, we cannot always get f.

Partition rows to cells: $C_1, C_2, \ldots, C_d$

Discrete f-approximators: $$\theta(C_i) = \frac{\sum_{\bar{x} \in S} \mu(\bar{x}) f(\bar{x})}{\mu(C_i)}$$

General f-approximators:

$$\left(x^{\langle 1 \rangle}, \ldots, x^{\langle k \rangle}, \gamma\right)$$



$$\theta(C_i) = \frac{\sum_{\bar{x}} \phi_i(\bar{x}) f(\bar{x})}{\mu(C_i)}$$

Assigning row to cells in a probablity of :

$$\phi_i(\bar{x}) / \mu(\bar{x})$$

Total measure of row
that is assigned to cell i.

Non-triviality

If a cell contains positive measures from 2 rows and their productivity functions are not equal (don’t just differ in group membership) , then they are non-trivial.

If an approximator has a non-trivial cell then it is non-trivial.

Simplicity

1. If 2 rows have some subfeatures that are the same, put them into the same cells. The index of these indices can be random $R \subseteq \{1, 2, \ldots, k+1\}$

1. Then we build a decision tree. Nodes could be $x^{\langle \ell \rangle} = b$
2. A cell is a **cube** if the members of a cell can be determined considering only the subset of the features.

   ⇒ Any discrete f-approximator with the above methods is a cube.

Simplicity

Definition: A simple f-approximator is a non-trivial discrete f-approximator for which each cell is a cube.

Admission Rules

Rank C_j , then take the first cells according to an admission rate r.

Equity: weighted average of the probability that an applicant from Group D is admitted.

Efficiency: weighted average of the productivity of the admitted.

Ideally, we want to maximize these.

Improvability and Maximality

If we cannot improve equity and efficiency for an approximator, then it is not improvable and is maximal.

⇒ Every trivial f-approximator is maximal.

Metrics - How do we define ‘strict improvability’?

- **Efficiency** is measured using the mean f-value of applicants admitted

$$V_h(r) \geq V_g(r) \forall r \in (0,1], \text{ and } \exists r \text{ s.t. } V_h(r) > V_g(r)$$

- **Equity** is measured using the fraction of D-applicants in the admitted group

$$W_h(r) \geq W_g(r) \forall r \in (0,1], \text{ and } \exists r \text{ s.t. } W_h(r) > W_g(r)$$

Result

Back to Problem 1: Pareto-Improvement!

- Pulling out rows of **high f-value associated with group D**,
  (or pull out rows of low f-values associated with group A)
- Simplest case:
  When all rows are in a single cell, we can always improve by separating out a row associated with group D with above-average f-value.
- Full proof that the operation above is always possible:
  shown for multiple cases, whose union constitutes ‘always’!

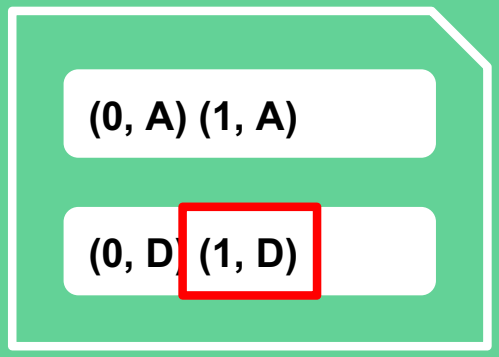
Back to Problem 1: Pareto-Improvement!

Simplest case

Group efficiency= 0.5
Group equity = 1/2

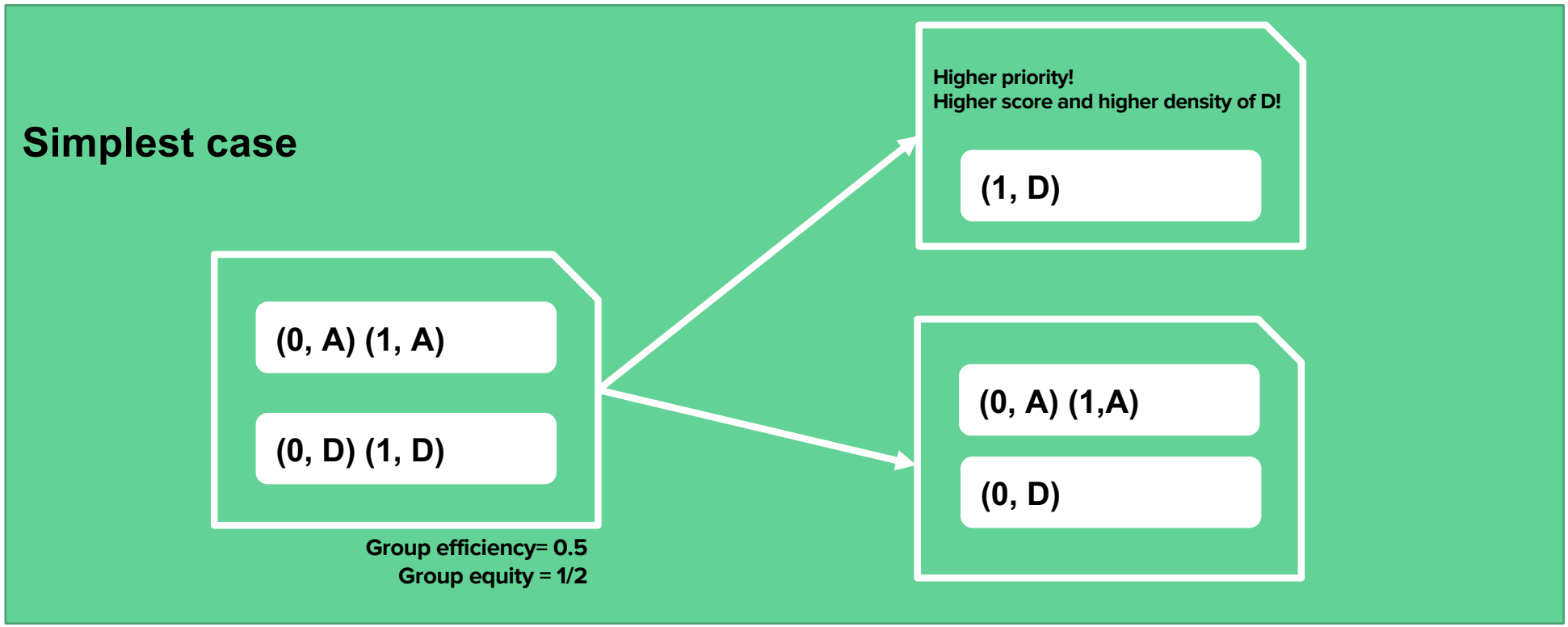
Back to Problem 1: Pareto-Improvement!

Simplest case

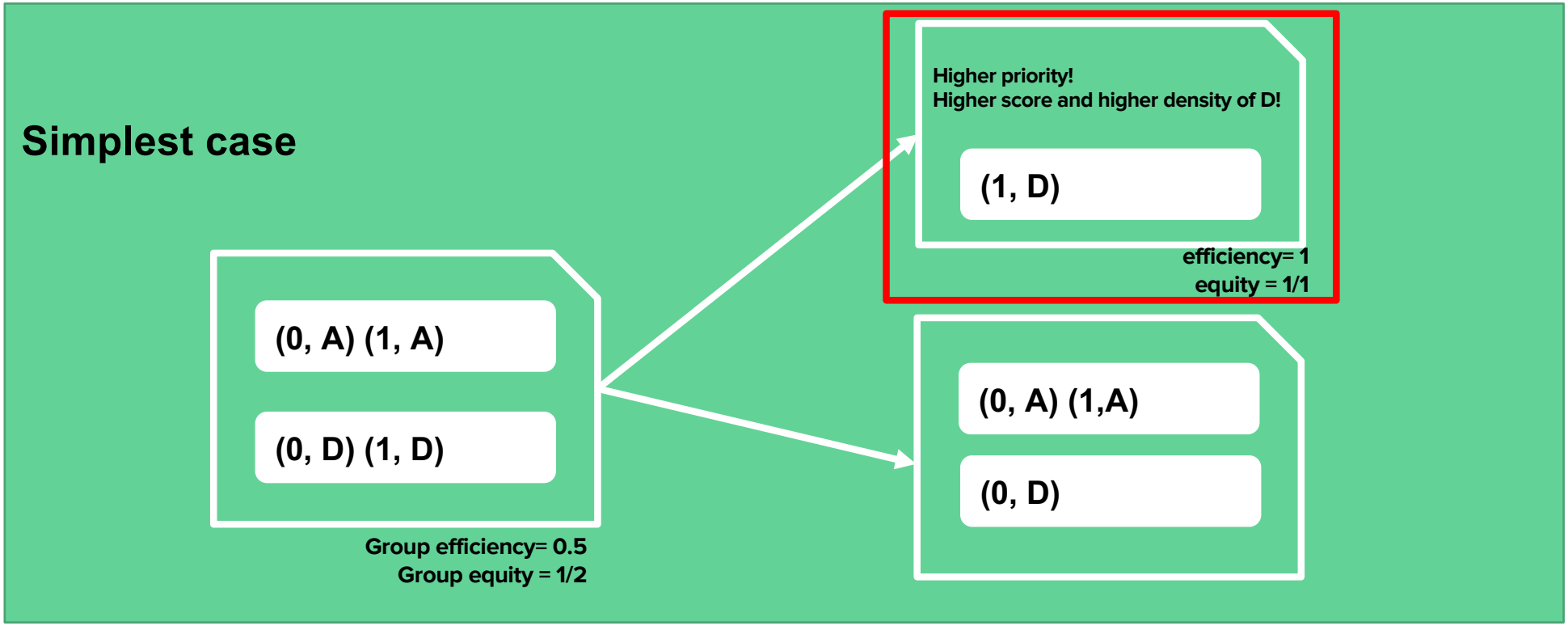


Group efficiency= 0.5
Group equity = 1/2

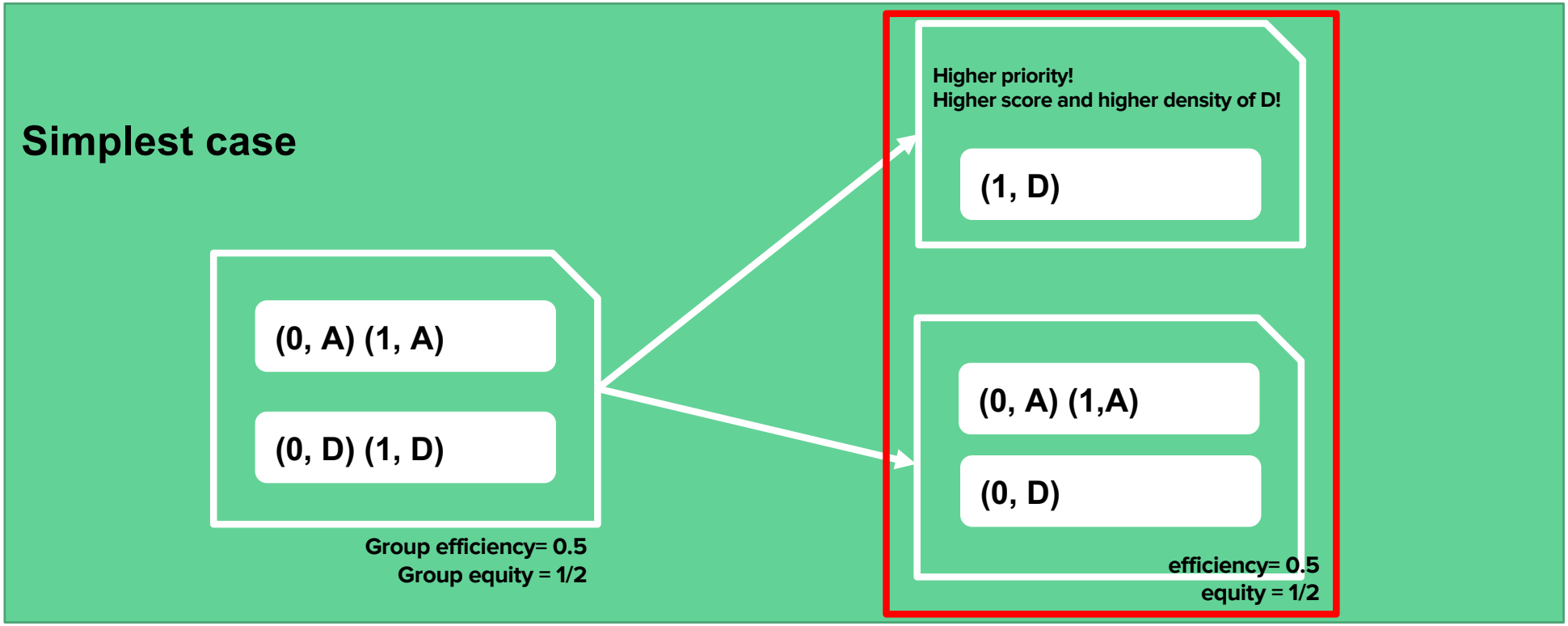
Back to Problem 1: Pareto-Improvement!



Back to Problem 1: Pareto-Improvement!



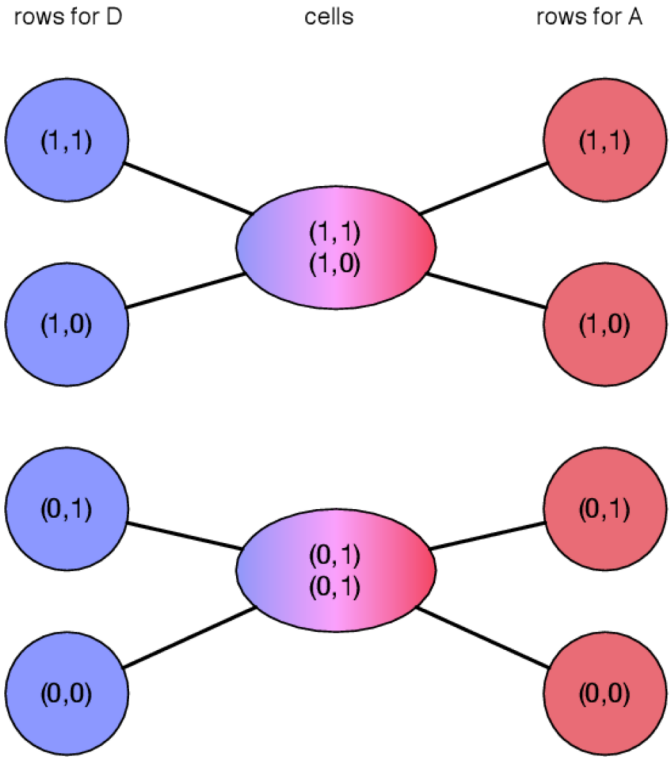
Back to Problem 1: Pareto-Improvement!



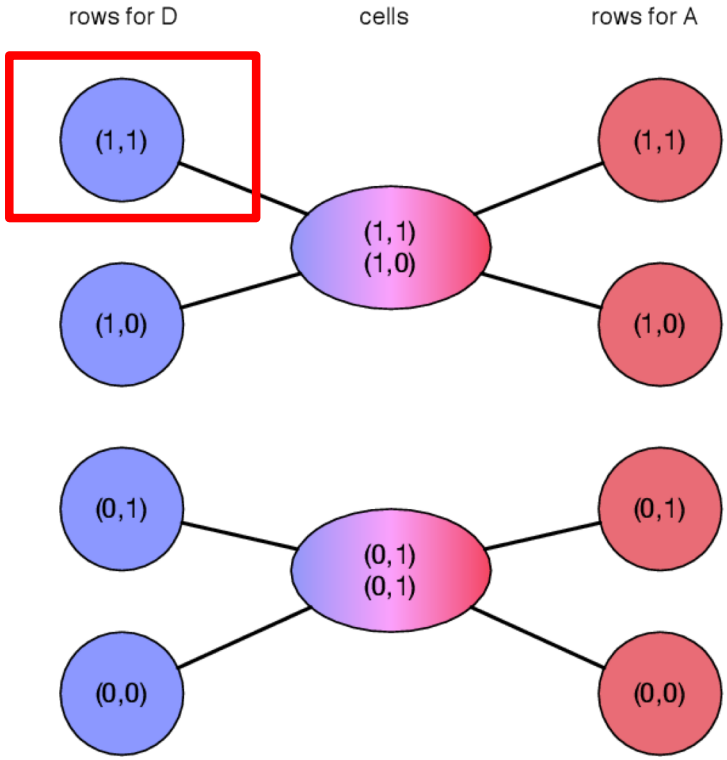
Back to Problem 1: Pareto-Improvement!

- Pulling out rows of **high f-value associated with group D**,
  (or pull out rows of low f-values associated with group A)
- Simplest case:
  When all rows are in a single cell, we can always improve by separating out a row associated with group D with above-average f-value.
- Full proof that the operation above is always possible:
  shown for multiple cases, whose union constitutes ‘always’!

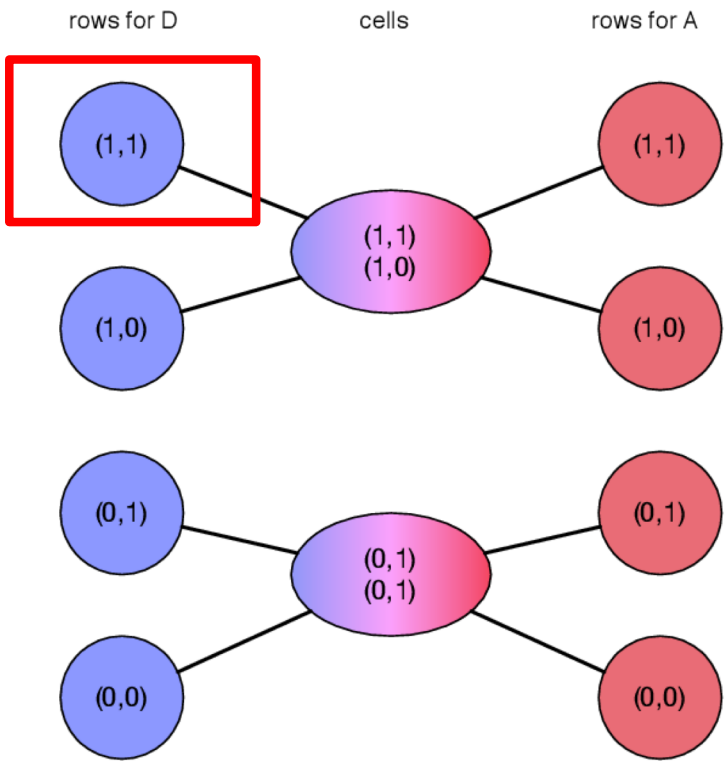
Quick Example

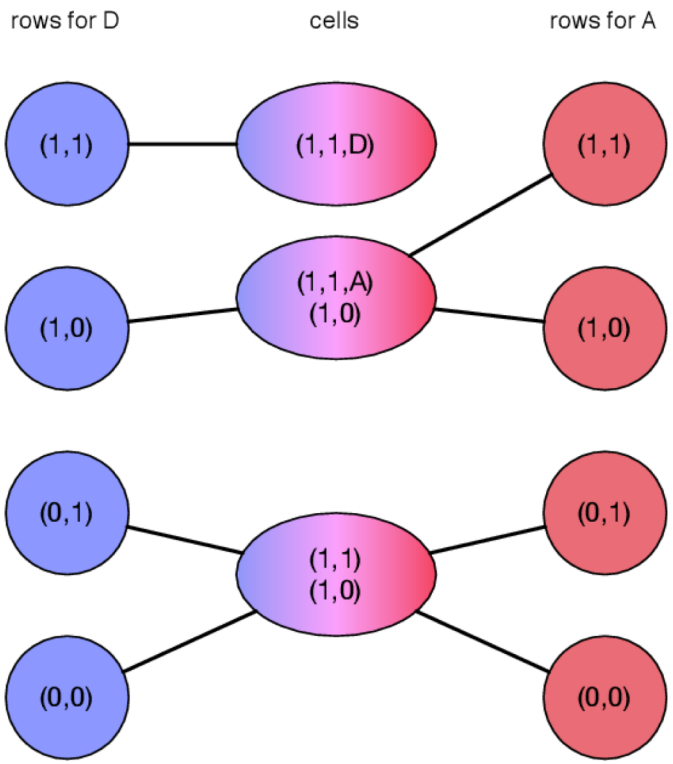


Quick Example

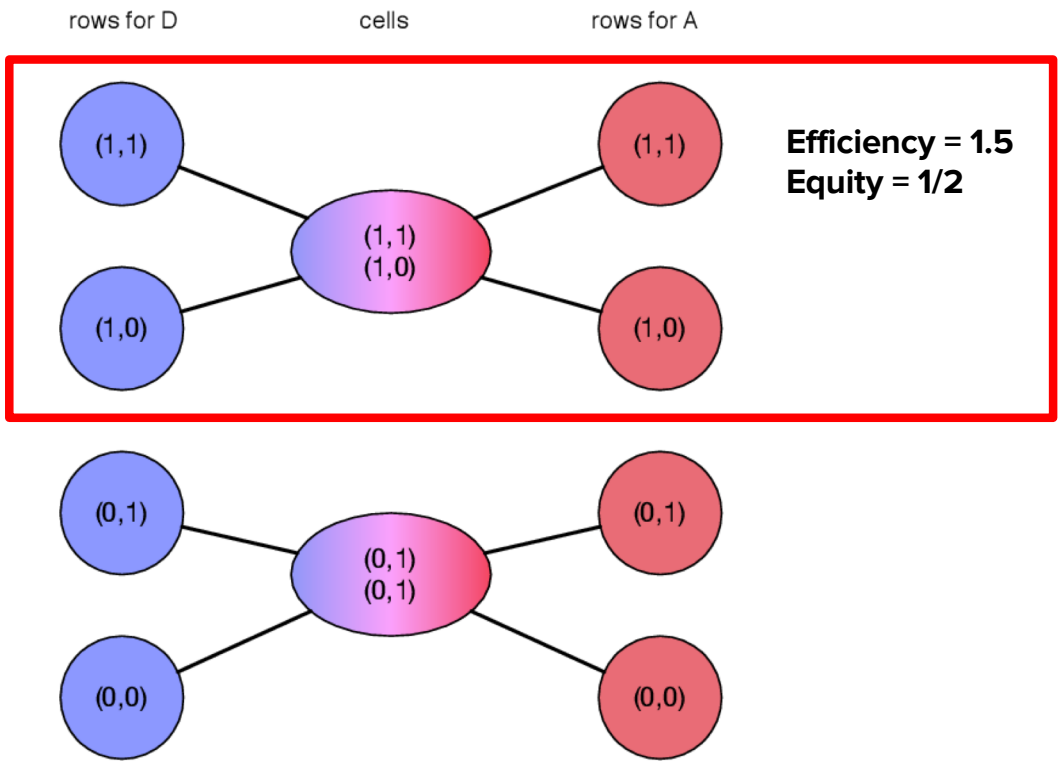


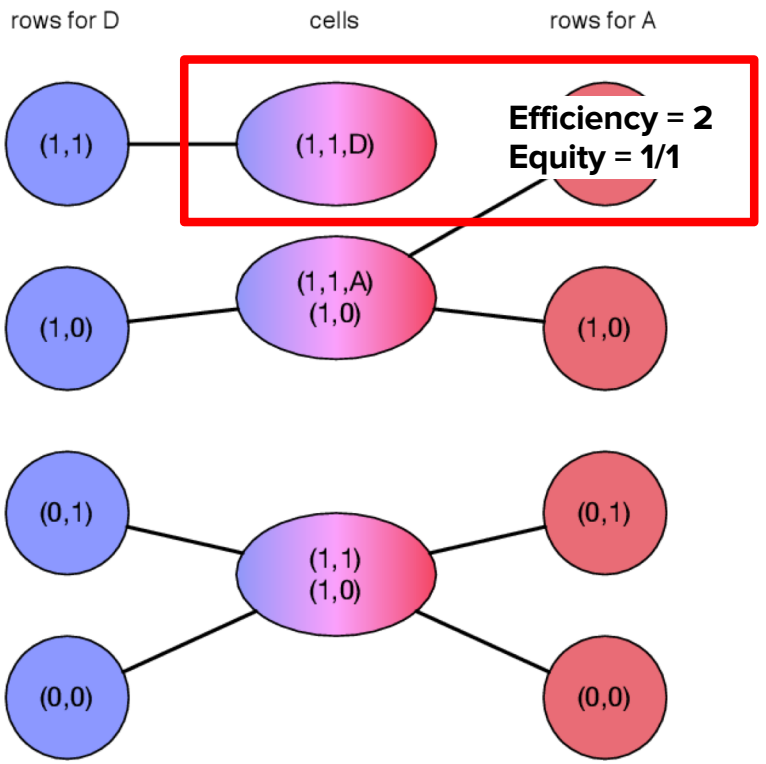
Quick Example



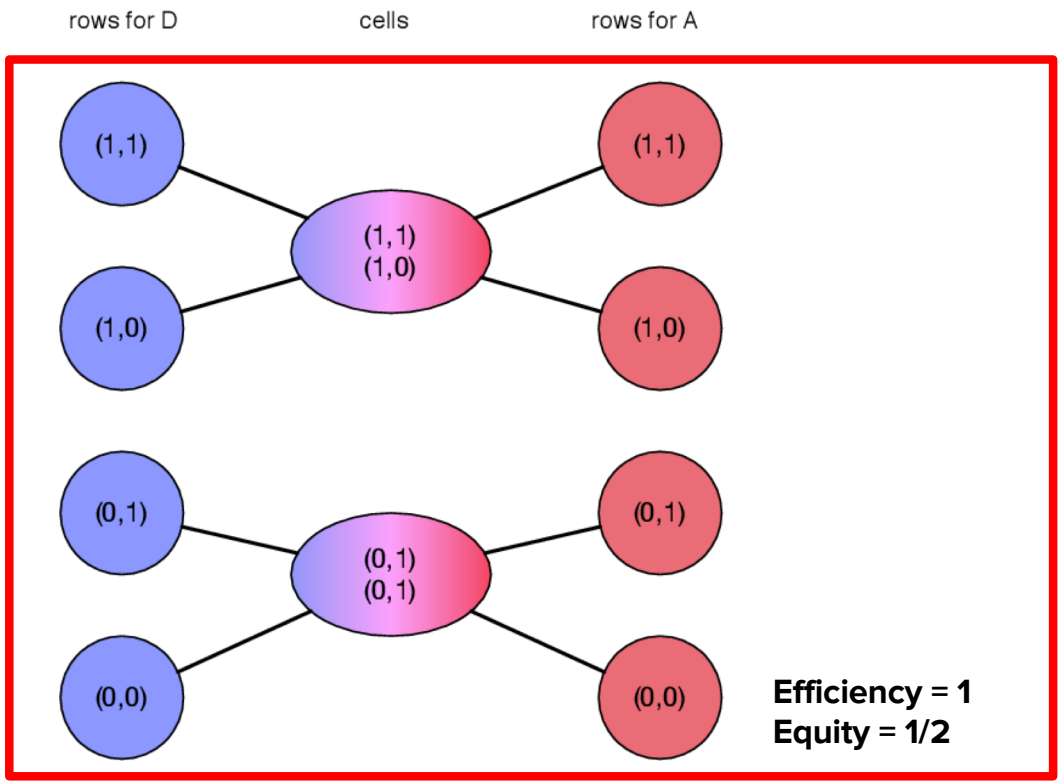


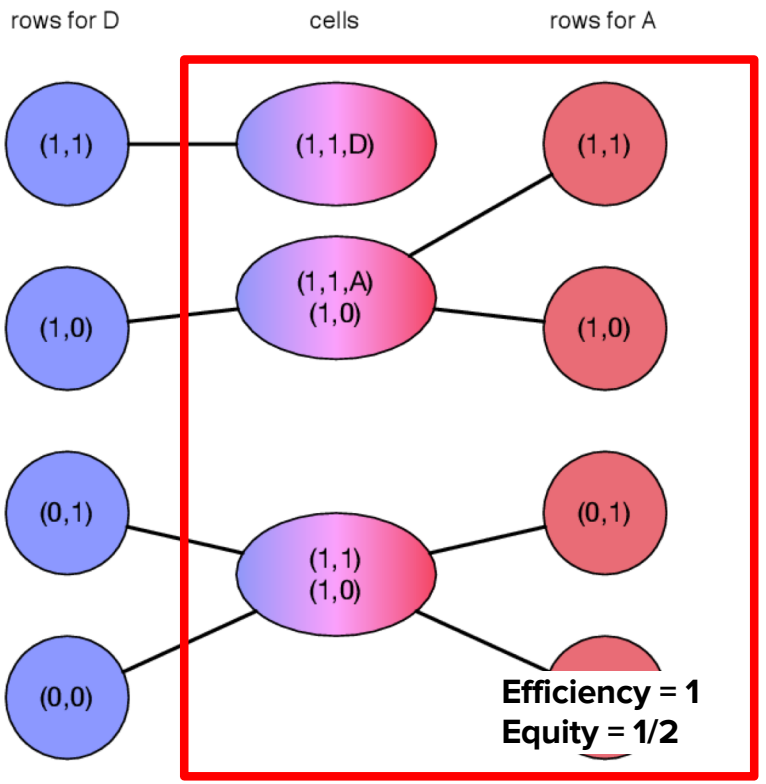
Quick Example





Quick Example





Back to Problem 2: Incentive Bias

- **Group-agnostic approximation transforms disadvantage to bias**
  Starting off from a simple, trival model, whenever a decision-maker can improve the efficiency by taking group membership into account, generates an incentive to use a rule that is explicitly biased in using group membership as part of the decision.

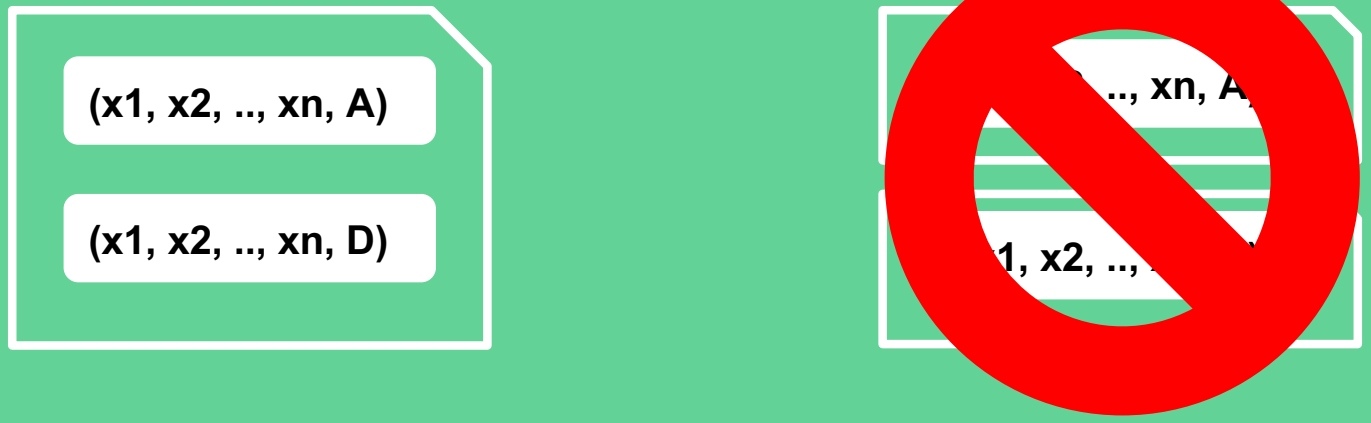
- Full proof:
  Shown by limiting our consideration to the improvability of non-trivial group-agnostic approximators

Back to Problem 2: Incentive Bias

Group-Agonistic approximator: (x,A) and (x,D) are always in the same cell!

Back to Problem 2: Incentive Bias

Group-Agonistic approximator: (x,A) and (x,D) are always in the same cell!

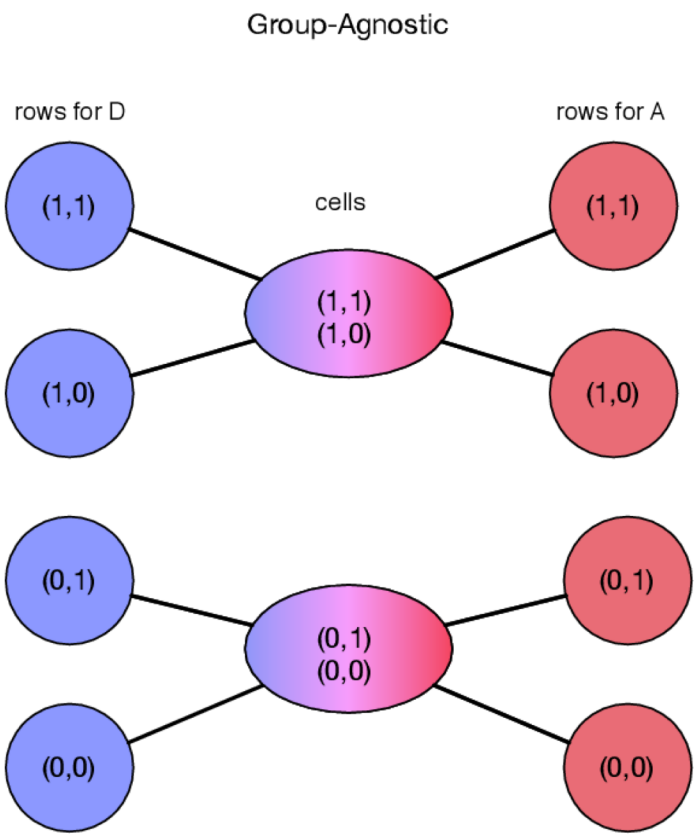


Back to Problem 2: Incentive Bias

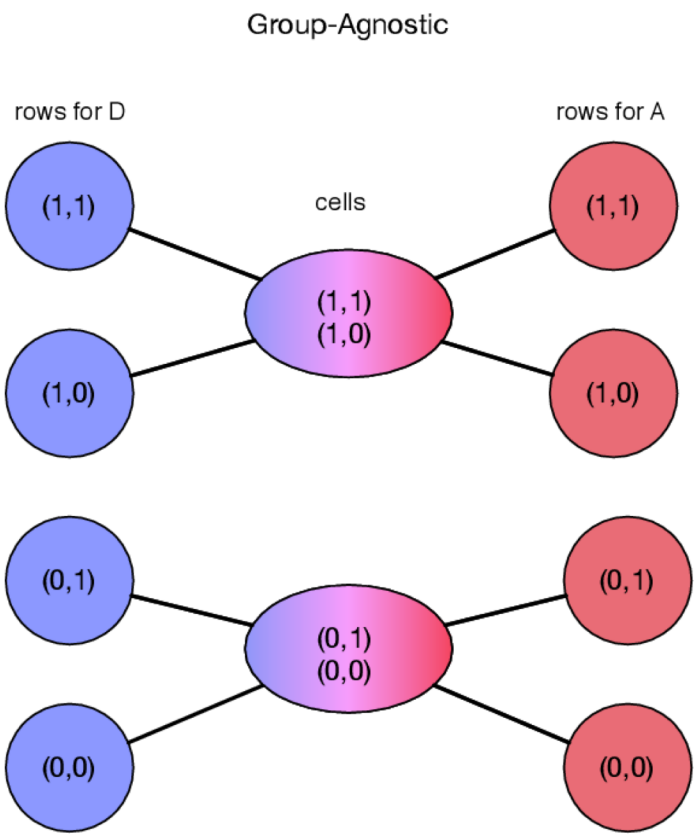
- **Group-agonistic approximation transforms disadvantage to bias**
  Starting off from a simple, trival model, whenever a decision-maker can improve the efficiency by taking group membership into account, generates an incentive to use a rule that is explicitly biased in using group membership as part of the decision.

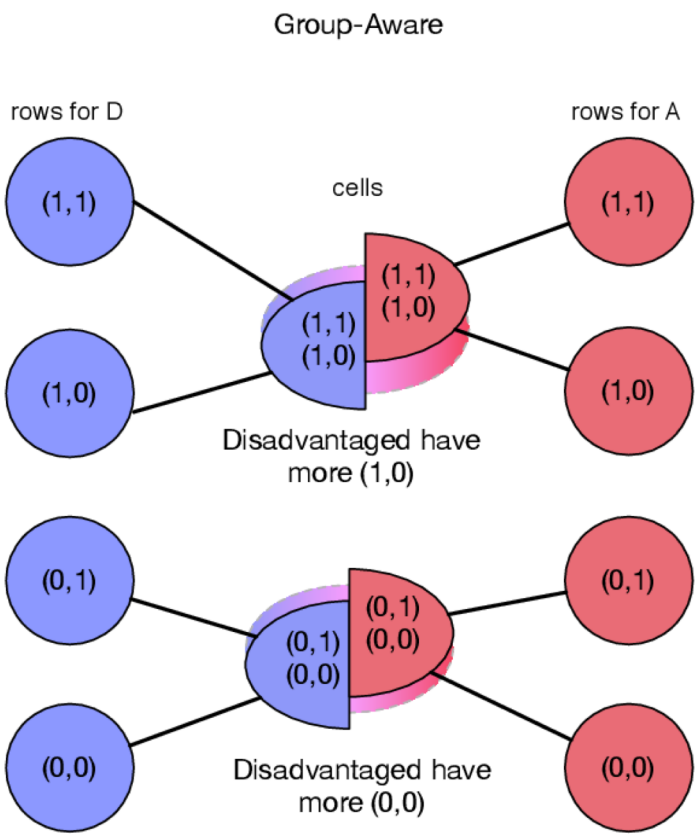
- Full proof:
  Shown by limiting our consideration to the improvability of non-trivial group-agonistic approximators

Quick Example



Quick Example





Discussion

Open Questions

- **Productivity:** what does it correspond to?

What is productivity?

“Each applicant has a productivity that is a **function of their feature vector**, and our **goal is to admit applicants of high productivity**. [...] we prefer applicants of higher productivity; [...] productivity can correspond to **whatever criterion determines the true desired rank-ordering** of applicants.”

- disadvantaged group has feature vectors resulting in lower productivity
- if productivity = what we truly care about:
  - D is “worse” at e.g. ability to perform a given job
- else: *productivity = proxy for what we truly care about*
  - Can we find a better proxy that increases both efficiency and equity?

Open Questions

- **Productivity:** what does it correspond to?
  - can we find a better criterion?
- **Genericity:** (limiting?) assumptions on productivity function

Genericity Assumption

Recall: Goal is to find *ordering* of candidates by productivity

- no “coincidental” equalities in f
- imposes discontinuity (or monotonicity, if continuous)
- hold trivially if we “think of all f-values as *perturbed by random real numbers* drawn independently from an arbitrarily small interval”

Genericity Assumption

Recall: Goal is to find *ordering* of candidates by productivity

- no “coincidental” equalities in f
- imposes discontinuity (or monotonicity, if continuous)
- hold trivially if we “think of all f-values as *perturbed by random real numbers* drawn independently from an arbitrarily small interval”

If genericity is due to random perturbation, is the ordering meaningful?

Are there cases where this doesn’t apply?

Open Questions

- **Productivity:** what does it correspond to?
  - can we find a better criterion?
- **Genericity:** (limiting?) assumptions on productivity function
  - random perturbations of productivity
- **Disadvantage:** can this condition be relaxed?

Disadvantage Condition

Suppose $f(x) > f(x')$ for some x, x'. Then $\frac{\mu(x, A)}{\mu(x, D)} > \frac{\mu(x', A)}{\mu(x', D)}$.

Requiring $f|U_A > f|U_D$ is not enough due to Simpson's Paradox.

Disadvantage Condition

Suppose $f(x) > f(x')$ for some x, x'. Then $\frac{\mu(x, A)}{\mu(x, D)} > \frac{\mu(x', A)}{\mu(x', D)}$.

When is this condition met in practice?

How can we bridge this gap?

Requiring $f|U_A > f|U_D$ is not enough due to Simpson's Paradox.

Open Questions

- **Productivity:** what does it correspond to?
  - can we find a better criterion?
- **Genericity:** (limiting?) assumptions on productivity function
  - random perturbations of productivity
- **Disadvantage:** can this condition be relaxed?
  - need to avoid Simpson's Paradox
- **Simplicity:** one notion of interpretability

Simplicity

Require: cells must be cubes (specify values of certain variables only)

Recall: applies to discrete f-approximators from variable selection or decision tree

Why simplify in the first place?

Are there any assumptions about the filtered out variables?

When does simplicity apply (and when not)?

Open Questions

- **Productivity:** what does it correspond to?
  - can we find a better criterion?
- **Genericity:** (limiting?) assumptions on productivity function
  - random perturbations of productivity
- **Disadvantage:** can this condition be relaxed?
  - need to avoid Simpson's Paradox
- **Simplicity:** one notion of interpretability
  - when does this definition apply?
- **Interpretability:** detecting biases vs. implying fairness

Interpretability

Tension:

helps **detect** bias and unfairness (Doshi-Velez & Kim)

vs.

implies fairness (Kleinberg & Mullainathan)

Interpretability

Tension:

helps **detect** bias and unfairness (Doshi-Velez & Kim)

vs.

implies fairness (Kleinberg & Mullainathan)

Also:

- interpretability vs. accuracy/efficiency?

Interpretability

Tension:

helps **detect** bias and unfairness (Doshi-Velez & Kim)

vs.

implies fairness (Kleinberg & Mullainathan)

Other reasons for interpretability (Rudin 2019):

- e.g. audit model for safety
- depending on value of interpretability, SPF could be optimal

Conclusion

- demonstrate relationship between simplicity and equity/efficiency within the many constraints of their framework
- constraints: maybe not realistic / don't generalize to real world problems

Generalizing:

- relax (some) of their assumptions and constraints
- how to trade off equity and efficiency with known preferences
- apply to real world problems and data
- extend to different problem besides admissions

Open Questions

- **Productivity:** what does it correspond to?
  - can we find a better criterion?
- **Genericity:** (limiting?) assumptions on productivity function
  - random perturbations of productivity
- **Disadvantage:** can this condition be relaxed?
  - need to avoid Simpson's Paradox
- **Simplicity:** one notion of interpretability
  - when does this definition apply?
- **Interpretability:** detecting biases vs. implying fairness
  - inherent value (other uses) of interpretability
- **Generalizations:** real world problems + data