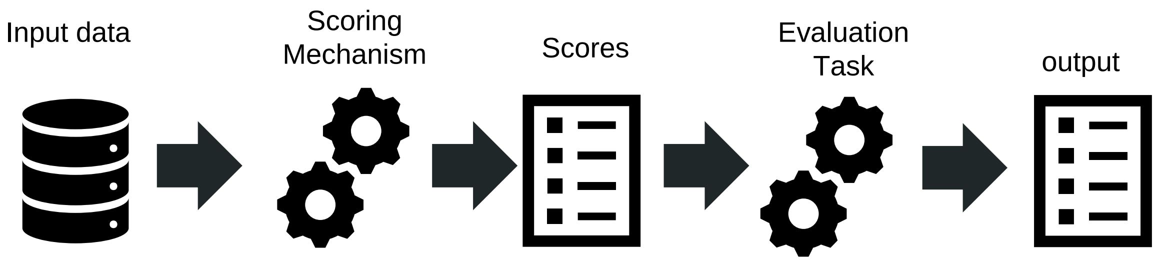
Score-based Evaluation

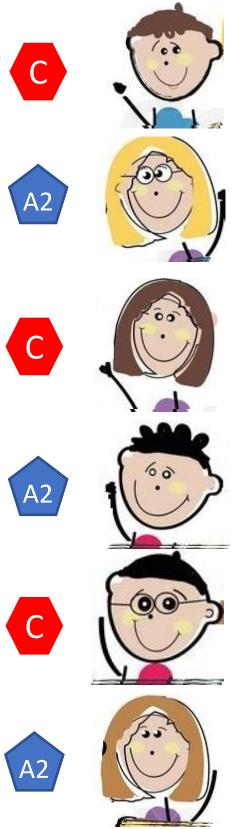
Score-based Evaluation



Toy Example

A2 Ann Arbor

D Chicago

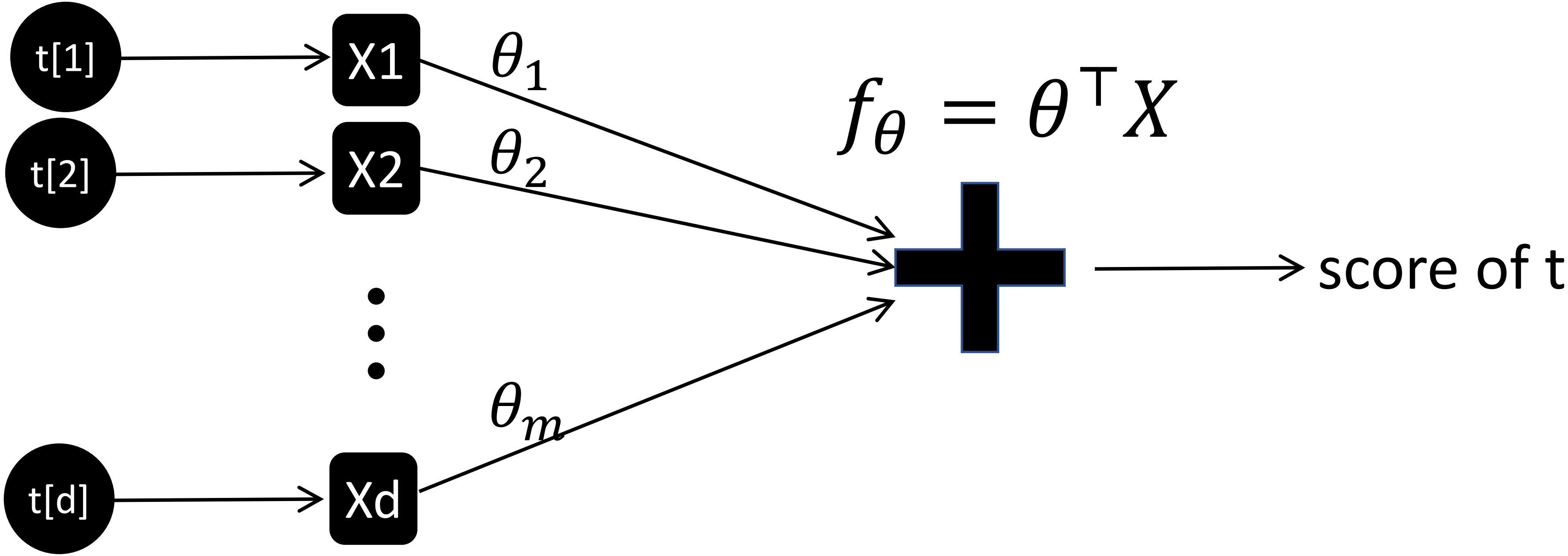


Suppose you own a real estate agency with two branches in Ann Arbor and Chicago.

You want to give bonus to

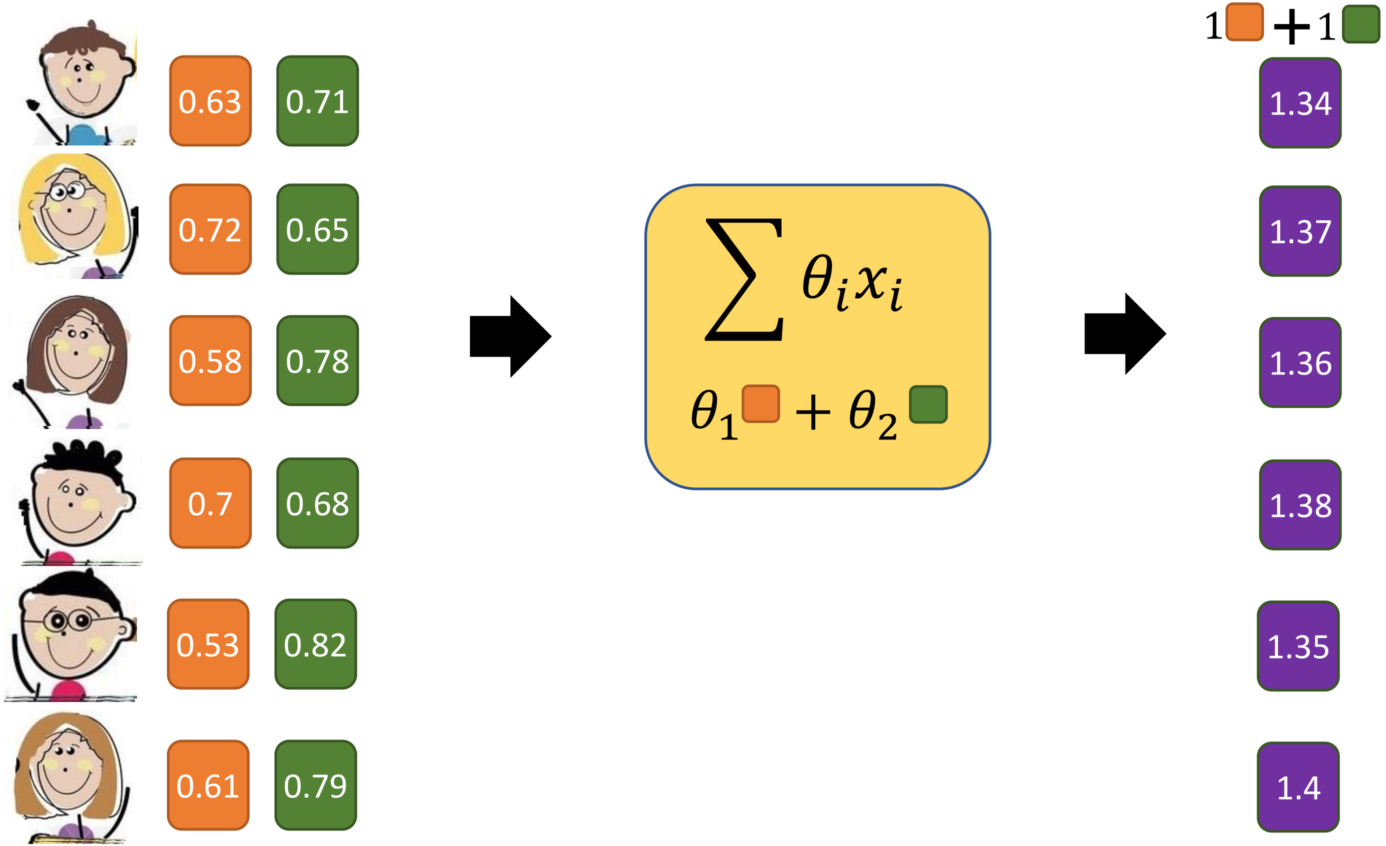
(1) Top-3 agents
(2) Successful agents

Scoring Mechanism: Linear Scoring



Toy Example

Sale -- Normalized
Customer Satisfaction -- Normalized



Converting non-linear to linear scoring

- Add non-linear terms as new attributes.
  - **Example:** $\mathrm{f} = 3X_1^2 + 5X_2^2 + X_1 + 2X_2$
  - **Set** $X_1' = X_1, X_2' = X_2, X_3' = X_1^2, X_4' = X_2^2$ **as the scoring attributes**
  - → $\mathrm{f} = 3X_3' + 5X_4' + X'_1 + 2X'_2$

- Use Log function to convert multiplication/exponential functions to linear
  - **Example:** $\mathrm{f} = 2^{X_1}.X_2^5$
  - **Set** $X_1' = X_1, X_2' = \log X_2$ **as the scoring attributes**
  - → $f' = \log f = (\log 2)\, X_1' + 5X_2'$

Ranking based on scoring

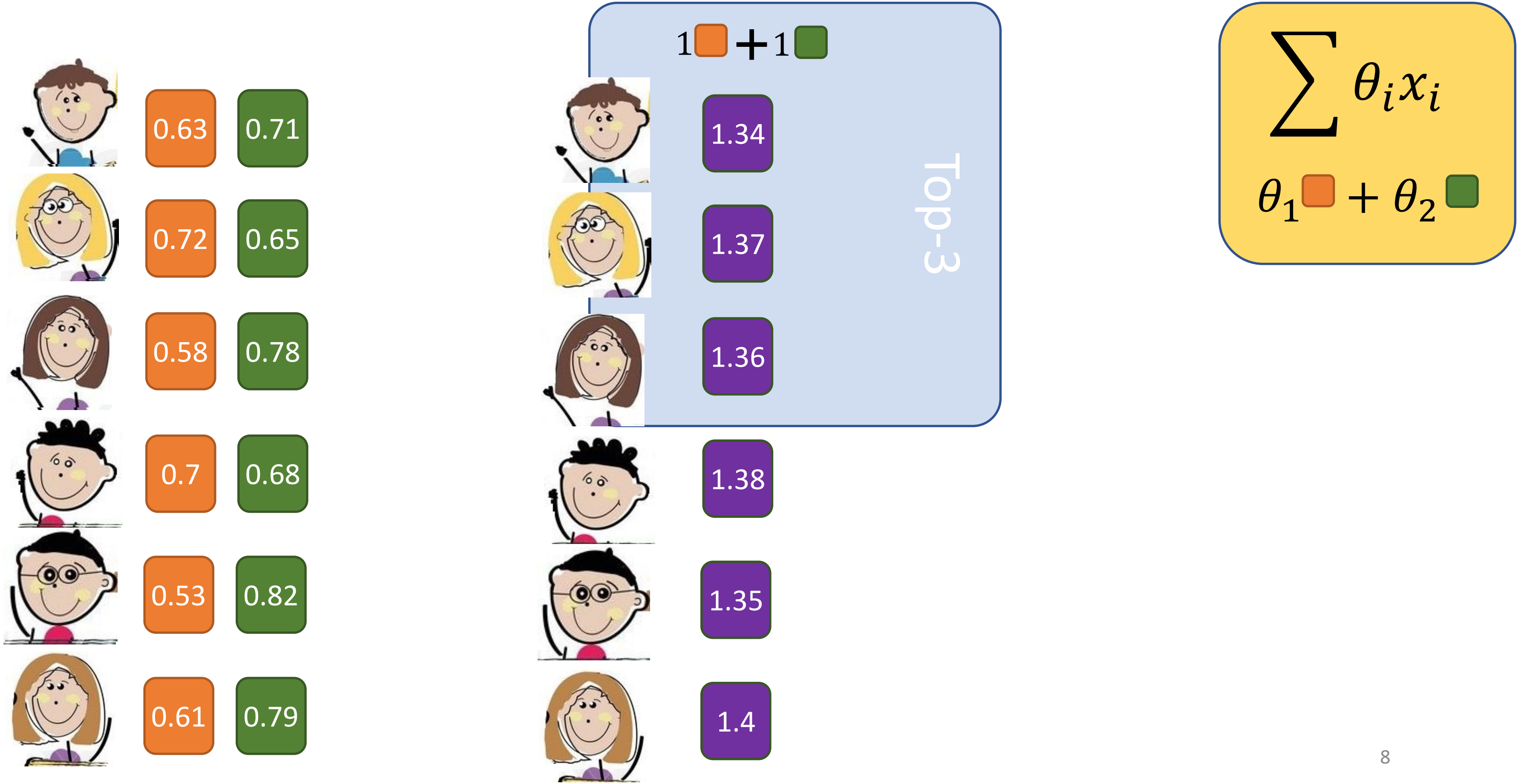
- Sort the scores to get the ranking

- (Select the top-k)

Toy Example

Sale -- Normalized

Customer Satisfaction -- Normalized



Classification based on scoring

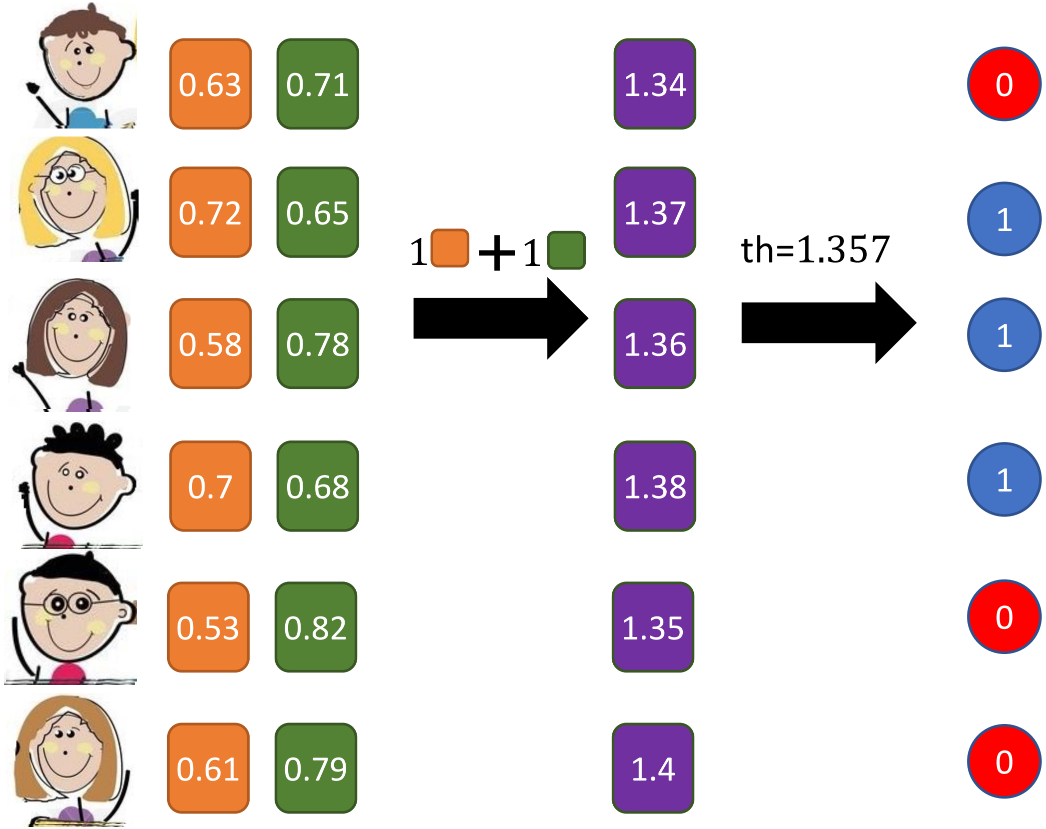
- Use score thresholds to specify decision boundaries

- For binary classification, for example, the scores above the threshold are classified as +1 (or accept) and the ones below it as -1 (or reject).

Toy Example

Sale -- Normalized

Customer Satisfaction -- Normalized



Machine learned scoring design

- Requires labeled training data (i.i.d sample from the underlying data distribution)
- Finds the parameter θ that minimizes the loss function $L(f)$

$$\min_{\theta} L(f_{\theta})$$

Human-designed scoring

- Human experts directly design the evaluator:
  - Ranking (no labeled training data) – e.g.: US News University Ranking
  - Classification – e.g.: RSA Scores
- “For predicting social outcomes, AI is not substantially better than manual scoring using a few features”[*]

[*] Narayanan, Arvind. "How to recognize AI snake oil." *Princeton University, Department of Computer Science,* (2019).

non-competitive v.s. competitive evaluation

- Non-competitive: the evaluation outcome for an entity only depends on the score of the entity itself (not others)
  - Example – classification: class label only depends on the score of an entity
- Competitive: the evaluation outcome depends also on the score of other entities being evaluated
  - Example – Ranking: The rank of an entity depends on the score of others

Responsible Scoring Interventions

Interventions to achieve responsible scoring

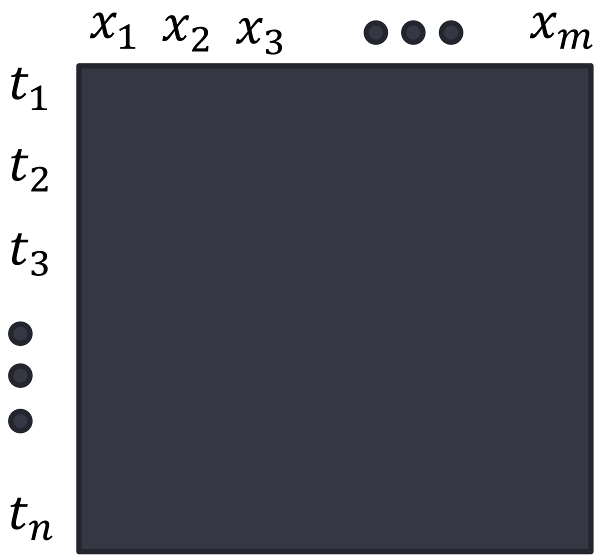
- Preprocess techniques
- Inprocess techniques (Scoring Algorithm Modification)
- Postprocess techniques

[*] S. A. Friedler, C. Scheidegger, S. Venkatasubramanian, S. Choudhary, E. P. Hamilton, and D. Roth. A comparative study of fairness-enhancing interventions in machine learning. In FAT*, 2019.

Pre-processing and Data Investigation

Reminder: Bias in rows v.s. columns

- Bias in rows: Not enough representative tuples from minority (sub)groups
- Bias in columns: Features are biased (correlated) with sensitive attributes



Data preprocessing techniques for classification without discrimination

Faisal Kamiran and Toon Calders

Knowledge and Information Systems 33.1 (2012): 1-33

- Preprocessing techniques for discrimination-free evaluation
    1. Suppression of Sensitive Attribute
    2. Massaging the dataset
    3. Reweighting
    4. Resampling
- Binary target variable, one binary sensitive attribute

Suppression of Sensitive Attribute

- To remove the attributes that highly correlate with the sensitive attribute.

Massaging the dataset

- Change the label of some tuples in the training data, in order to minimize the discrimination.
- Considers a subset of data from the minority group as promotion candidates:
  - Change the labels of promotion candidates from – to +
- a subset of data from the majority group as demotion candidates:
  - Change the labels of demotion candidate from + to –
- Which labels to select?
      - Learn a classifier; rank the tuples based on their probability of having positive labels
      - Select the top-k of minority (for promotion) and the bottom-k of majority (for demotion)

Reweighting

- Instead of changing the labels, each tuple in the training data is assigned with a weight

- This works for all the methods for which tuple weights can be used as frequency counts

1. For each of the group-value combinations, it computes the probability if independence would hold.
2. The weight of a group is ratio b/w its probability under independence and it actual probability in the dataset

Reweighting, Example

$$P_{exp}(sex = f \wedge X(class) = +) = .5 \times .6 = \ .3$$

From the dataset:

$$P(sex = f \wedge X(class) = +) = \ .2$$

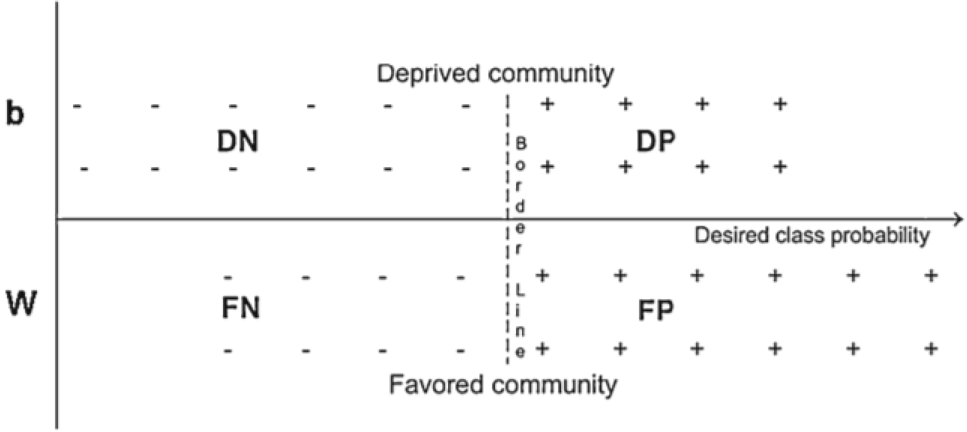
→ $W(x) = {}^{0.3}/_{0.2} = 1.5$

Sex	Ethnicity	Highest degree	Job type	Class
M	Native	H. school	Board	+
M	Native	Univ.	Board	+
M	Native	H. school	Board	+
M	Non-nat.	H. school	Healthcare	+
M	Non-nat.	Univ.	Healthcare	–
F	Non-nat.	Univ.	Education	–
F	Native	H. school	Education	–
F	Native	None	Healthcare	+
F	Non-nat.	Univ.	Education	–
F	Native	H. school	Board	+

Resampling

- Calculate the sample size for each of the group-value combination.
    - e.g.: {male reject, male accept, female reject, female accept}

Sample size	DP	DN	FP	FN
Actual	8	12	12	8
Expected	10	10	10	10



Optimized pre-processing for discrimination prevention

Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R. Varshney

Advances in Neural Information Processing Systems. 2017.

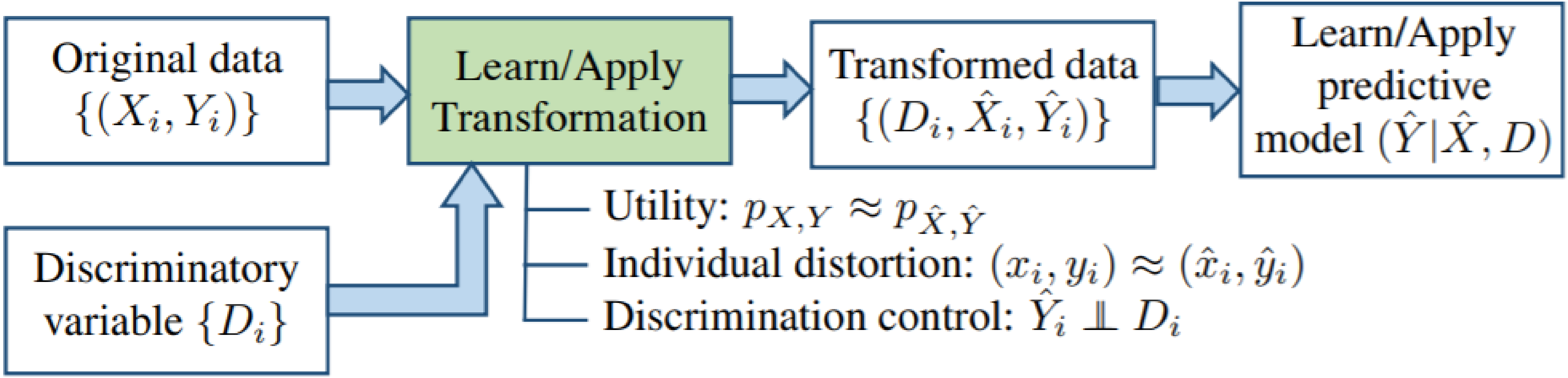
- A probabilistic formulation of data pre-processing to reduce discrimination

- Convex optimization to learn a data transformation that:

1. Control discrimination

2. Limit the distortion in individual data samples

3. Preserve utility


Original data $\{(X_i, Y_i)\}$
Learn/Apply Transformation
Transformed data $\{(D_i, \hat{X}_i, \hat{Y}_i)\}$
Learn/Apply predictive model $(\hat{Y}|\hat{X}, D)$
Discriminatory variable $\{D_i\}$
Utility: $p_{X,Y} \approx p_{\hat{X},\hat{Y}}$
Individual distortion: $(x_i, y_i) \approx (\hat{x}_i, \hat{y}_i)$
Discrimination control: $\hat{Y}_i \perp\!\!\!\perp D_i$

$$\min_{p_{\hat{X},\hat{Y}|X,Y,D}} \Delta\left(p_{\hat{X},\hat{Y}}, p_{X,Y}\right)$$

Utility Preservation

$$\text{s.t.}\quad J\left(p_{\hat{Y}|D}(y|d), p_{Y_T}(y)\right) \leq \epsilon_{y,d} \text{ and}$$

Discrimination Control

$$\mathbb{E}\left[\delta((x,y),(\hat{X},\hat{Y})) \mid D=d, X=x, Y=y\right] \leq c_{d,x,y}\ \forall\,(d,x,y) \in \mathcal{D}\times\mathcal{X}\times\mathcal{Y},$$

$$p_{\hat{X},\hat{Y}|X,Y,D} \text{ is a valid distribution.}$$

Individual Distortion Control

$$J(p,q) = \left|\frac{p}{q} - 1\right|$$

Certifying and removing disparate impact

Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian

KDD 2015

- The goal is to certify and remove disparate impact by modifying each attribute such that:

   1. predictability of sensitive attribute using the input data is impossible (minimized)

   2. predictability of class label is preserved

Disparate Impact

- Consider an attribute *X*, a single binary sensitive attribute *S*, and a binary classifier *f*

- *f* has disparate impact of *t*, if:

$$\frac{P(f(X) = 1|S = 0)}{P(f(X) = 1|S = 1)} \leq t$$

- That is, the probability that a member of a protected class being classified as 1 (accept) is at most *t* times (e.g. t=80% -- the 80% rule) less than a member of unprotected class.

Certifying disparate impact

- The main idea is that a classifier $f(X)$ **does not have disparate impact, if the sensitive attribute *S* is not predictable by *X*.**

- → We can check the data without knowing the label attribute or the even the algorithm

Certifying Disparate Impact

- **Balanced Error Rate (BER): consider a classifier $g: X \to S$**

$$BER(g(X), S) = \frac{P(g(X) = 0|S = 1) + P(g(X) = 1|S = 0)}{2}$$

- **ϵ-Predictability: The data is ϵ-predictable if there exists $g: X \to S$ such that $BER(g(X), S) \leq \epsilon$**

Theorem: If a dataset D admits a classifier f with disparate impact of 0.8, then D is $(0.5 - \frac{B}{8})$-predictable, where $B = P(F(X) = 1|S = 0)$

$$BER(f(X), S) = \frac{P(f(X) = 0|S = 1) + P(f(X) = 1|S = 0)}{2}$$

$$= \frac{1 - P\big(f(X) = 1\big|S = 1\big) + B}{2}$$

$$\leq \frac{1 - P\big(f(X) = 1\big|S = 0\big)/0.8 + B}{2}$$

$$= \frac{1 - B/0.8 + B}{2} = \frac{1}{2} - \frac{B}{8}$$

Removing Disparate Impact

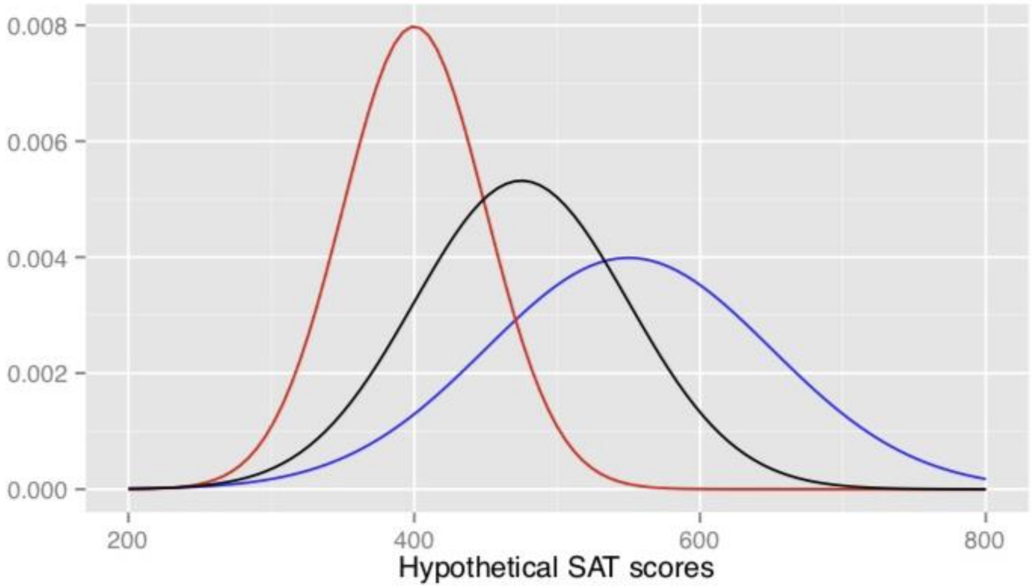
- It is easy to remove the data disparate-impact free: Just set all values of X'=0
- This, however, removes the power of data to predict class labels
- We want to transform X to X' such that prediction power of data is preserved:
    - we want to transform X in a way that the rankings within demographic groups is preserved (but not necessarily across groups).

Removing Disparate Impact

- Let p_x^s be the percentage of tuples at group $S = s$ with value at most $X = x$

- for each tuple (x_i, s_i):
  - Calculate $p_{x_i}^{s_i}$
  - Find x_i^{-1} such that $p_{x_i^{-1}}^{(1-S_i)} = p_{x_i}^{s_i}$
  - Repair $\overline{x_i}$ as median (x_i, x_i^{-1})

Removing Disparate Impact



Interventional Fairness: Causal Database Repair for Algorithmic Fairness

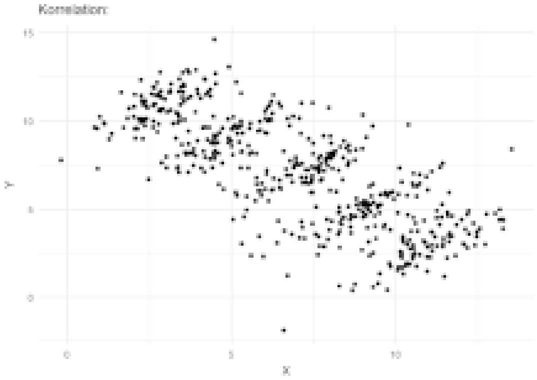
Babak Salimi, Luke Rodriguez, Bill Howe, Dan Suciu

SIGMOD 2019

- Repair the pre-existing human bias before using the data for learning
- Proposes the causal notion of fairness and reduces the problem to dataset repair

Associational Fairness can be misleading

- Simpson's Paradox
  - e.g.: UC Berkeley’s 1973 Gender Bias case



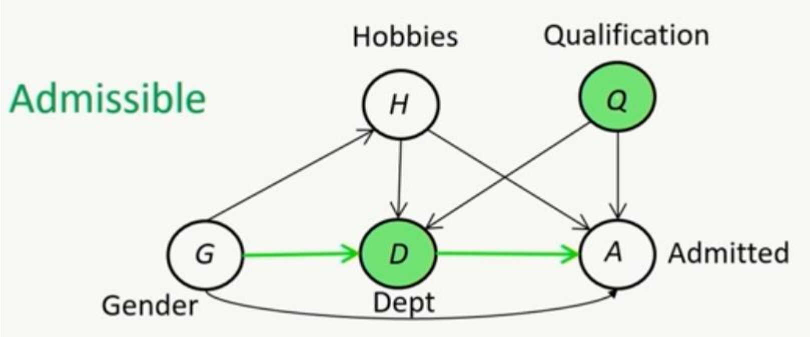
	Men		Women	
	Applicants	Admitted	Applicants	Admitted
Total	8442	**44%**	4321	35%

Department	Men		Women	
	Applicants	Admitted	Applicants	Admitted
A	***825***	62%	108	**82%**
B	***560***	63%	25	**68%**
C	325	**37%**	***593***	34%
D	417	33%	375	**35%**
E	191	**28%**	***393***	24%
F	373	6%	341	**7%**

* Image and data are taken from Wikipedia

- User specify admissible variables K, only allow causal influence through K

- Admissible variables are socially not discriminative



- An application is fair if the protected attribute does not affect the outcome for any possible configuration of admissible variables

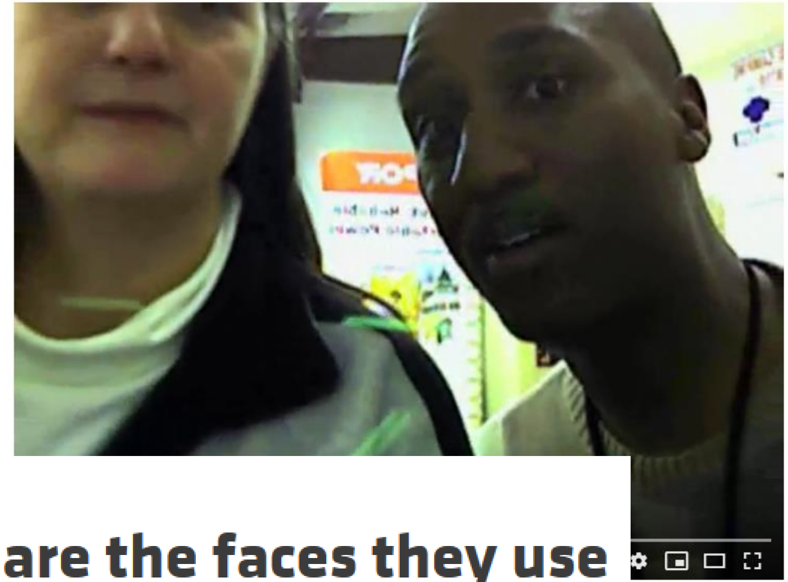
- Given admissible variables, derive a set of conditional independence constraints that imply interventional fairness.
- Model as a database repair problem, get free algorithms
- Classifiers trained on repaired data:
  - Provably fair by interventional fairness
  - Empirically fair by other metrics

Assessing and Remedying Coverage for a Given Dataset

A. Asudeh, Z. Jin, H. V. Jagadish

ICDE 2019

Motivation



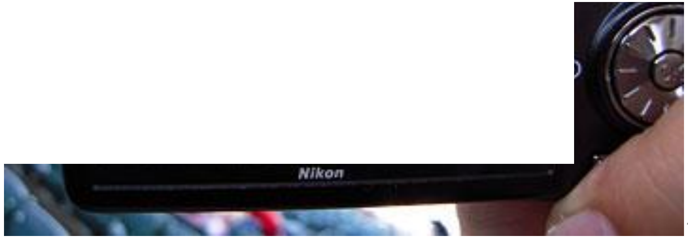
- Google Gorilla
- Nikon camera’s open eyes detection
- The face tracking feature of the HP web cams

ARTIFICIAL INTELLIGENCE DIVERSITY

Most engineers are white — and so are the faces they use to train software

A black researcher had to wear a white mask to test her own project.

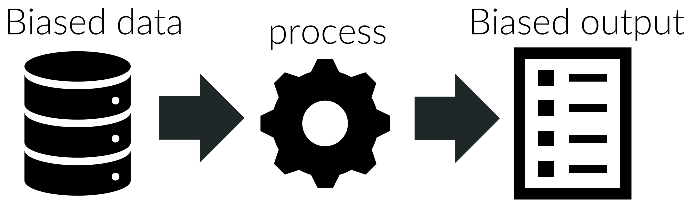
By Tess Townsend | Jan 18, 2017, 11:45am EST



racism in, racism out!

- In these cases, it is the data that causes the issue!

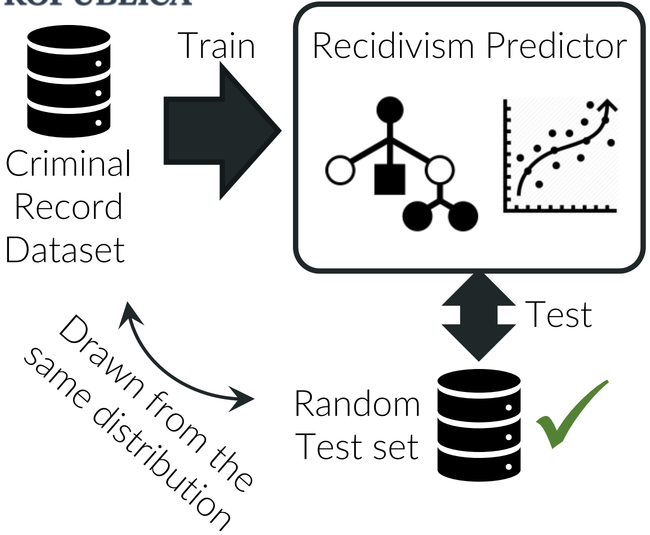
- Lack of “Coverage”: Not having enough representatives from the minority subgroups

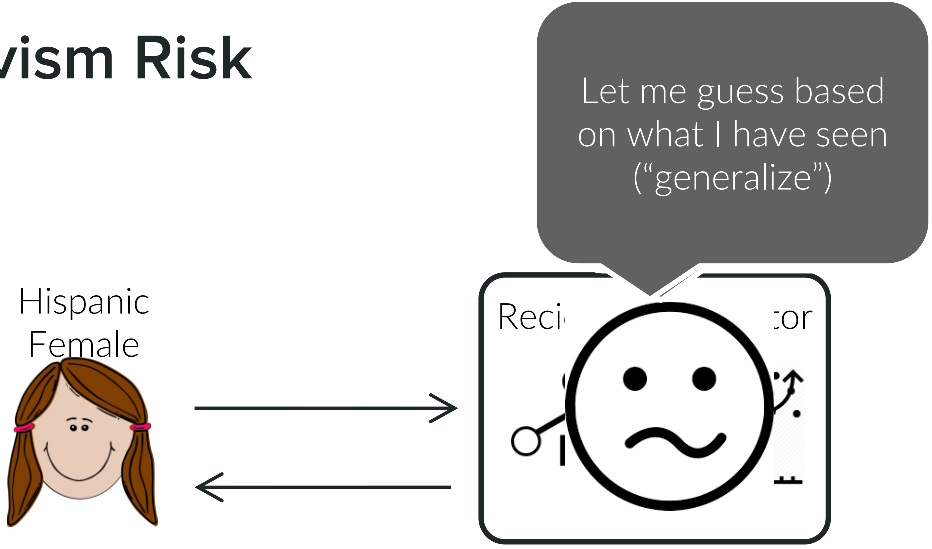


- Lack of “Coverage”: Not having enough representatives from the minority subgroups

Example: predicting the recidivism Risk

PROPUBLICA





(Lucky): Similar "behavior" →

(Unlucky): Diff. "behavior" →

Identifying lack of coverage

- Our Scope: Low-cardinality categorical attributes

- Pattern: A vector of size #attributes, in which P[i] is either a fixed value or is unspecified (i.e. X)
  - e.g. X2X0 is a pattern: all tuples where A_2=2 and A_4=0 match it
- Parent/child relation
  - P_i is parent of P_j, if it replaces a det. cell of P_j with X
    - e.g. X2XX is a parent of X2X0
  - Parents are more general: more value combinations match them

Problem 1: Max. Uncovered Pattern (MUP) Identification

- Uncovered Pattern
  - A patterns that #tuples matching it is less than a threshold τ
- Maximal Uncovered Pattern (MUP)
  - An uncovered pattern that all of its parents are covered

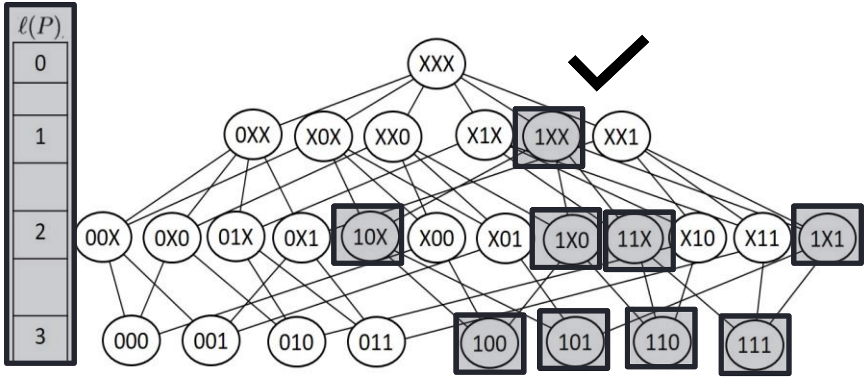
- Problem1: find all MUPs
  - Theorem1: No polynomial time alg. exists for Problem1

Pattern Graph

- Nodes: pattern
- Edges: b/w parent child patterns
- Level of a node: #deterministic cells
- In this example:
  - 101 is uncovered: no tuple matches it
  - It is not a MUP: Its parent 1X1 is also uncovered
  - There is only one MUP: 1XX
    - → Its parent (XXX) is covered

Example (a dataset with 3 binary attributes, τ=1)

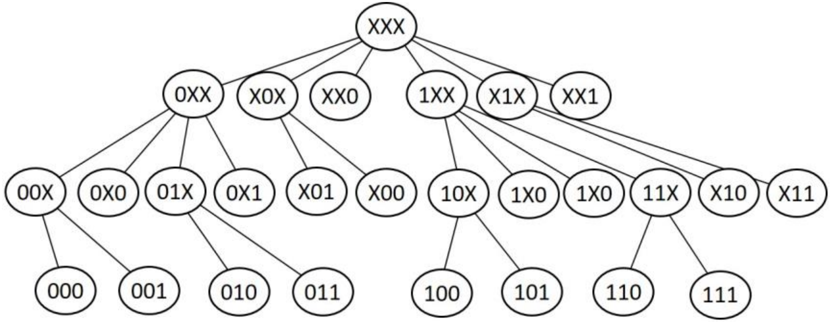
	A1	A2	A3
t1	0	1	0
t2	0	0	1
t3	0	0	0
t4	0	1	1
t5	0	0	1

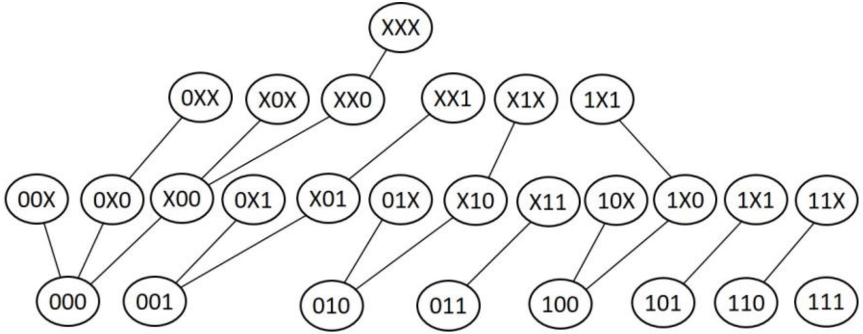


Identifying Lack of Coverage

Summary of Techniques

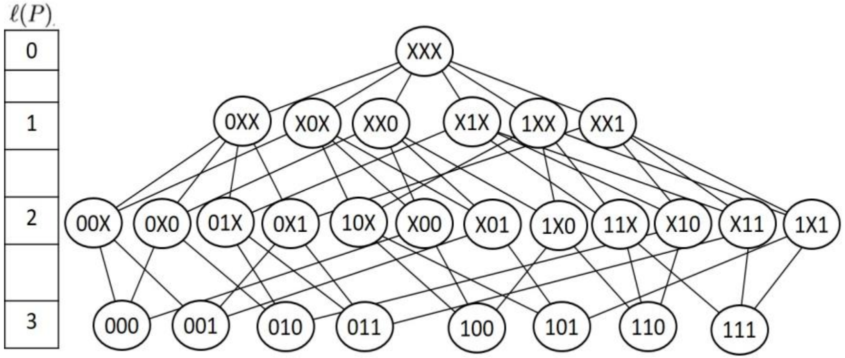
- *PATTERN-BREAKER* **(Top-Down BFS):**
  - Rule1: transfers the graph to a tree that guarantees generation of each MUP once
  - not efficient if MUPs are at the bottom
- *PATTERN-COMBINER* **(Bottom-up BFS):**
  - Rule 2: transfers the graph to a forest
  - not efficient if MUPs are at the top
- *DEEPDIVER* **(Fast DFS Space Pruner):**
  1. Applying DFS while following Rule1, quickly find an uncovered node
  2. Change the direction upward to find a MUP
  3. prune <u>both ancestors and descendants of MUPs</u>





Coverage Enhancement

Human-in-the-loop is necessary to set up the oracle for marking out the invalid MUPs



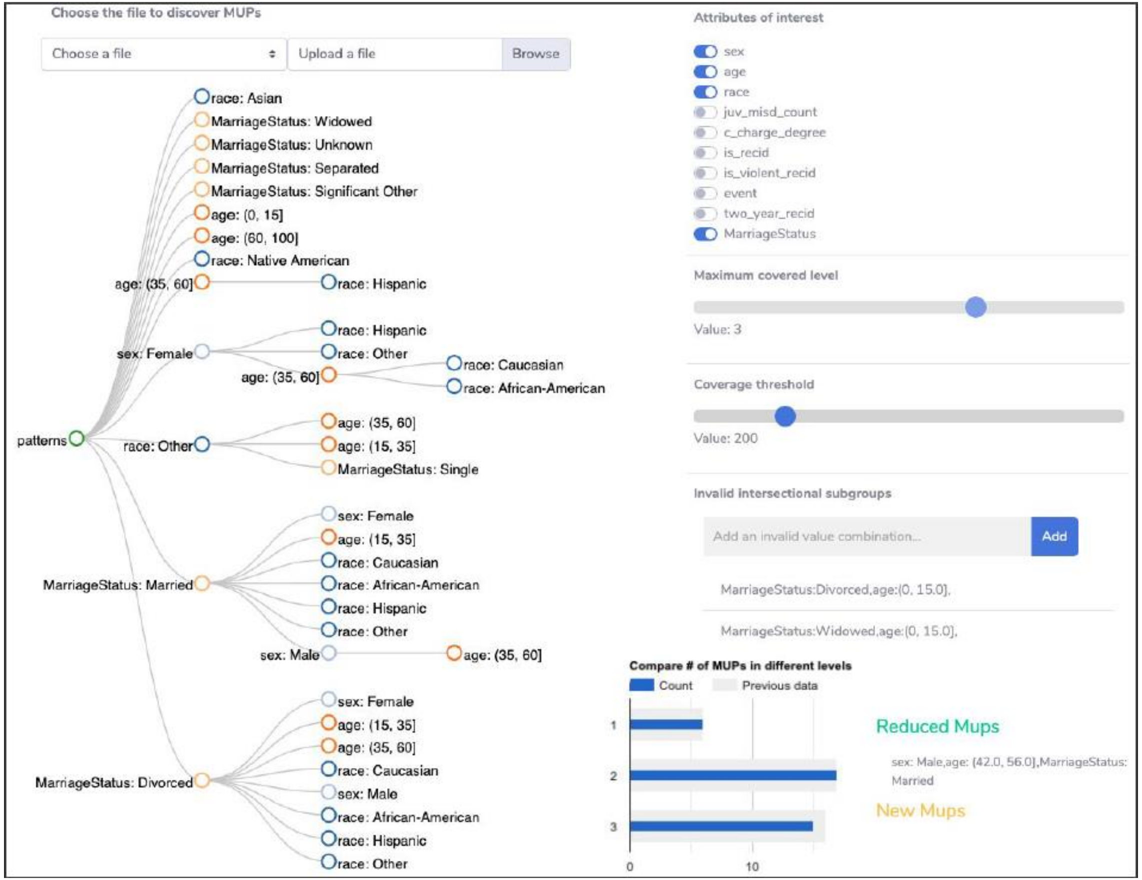
- Question: What is the minimum #tuples to collect to make sure there is no MUP on or above a certain level $\boldsymbol{\ell}$?
  - NP-hard (reduction from vertex-cover problem)
  - Modeled the problem as *hitting set*
    - Items: value combinations
    - Sets: uncovered patterns at level $\boldsymbol{\ell}$
  - Efficient implementation of the greedy algorithm is the challenge
    - Designed proper inverted indices and a tree data structure

Coverage over Linked Data

- To efficiently identifying coverage where attributes of interest are scattered over multiple relations:
  - Baseline: join all the tables and then study coverage
    - Not practical due to the time/space complexity
- Solutions:
  - Indexing schema to speed up the COUNT query
  - Priority based algorithm to reduce the number of COUNT operations
  - Approximate coverage analysis

[*] Yin Lin, Yifan Guan, Abolfazl Asudeh, and H. V. Jagadish. Identifying Insufficient Coverage of Databases with Multiple Relations, VLDB, 2020.

MithraCoverage



[*] Z Jin, M Xu, C Sun, A Asudeh, and H. V. Jagadish. MithraCoverage: A System for Investigating Population Bias for Intersectional Fairness. In SIGMOD 2020.

Scoring Design and Algorithm Modification

Classification

Reminder

- Finds the parameter θ that minimizes the loss function $L(f)$

$$\min_{\theta} L(f_\theta)$$

- For efficient learning, the loss function is designed to be convex
- Optimizing the loss function, without considering demographic groups may result in “unfair” models
- Changing the problem formulation to account for fairness

$$\min_{\theta} \quad L(f_\theta)$$
$$s.t. fairness$$

- Challenge: This is (often) not convex

Adding fairness makes the optimization non-convex

- e.g.:
  - $\min L(\theta)$
    - s.t. $P(f_\theta(X) = 1|S = 0) = P(f_\theta(X) = 1|S = 1)$ Demographic Parity
  - $\min L(\theta)$
    - s.t. $P(f_\theta(X) \neq y|S = 0) = P(f_\theta(X) \neq y|S = 1)$ Misclassification Partiy

Fairness constraints: Mechanisms for fair classification

Muhammad Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P. Gummadi

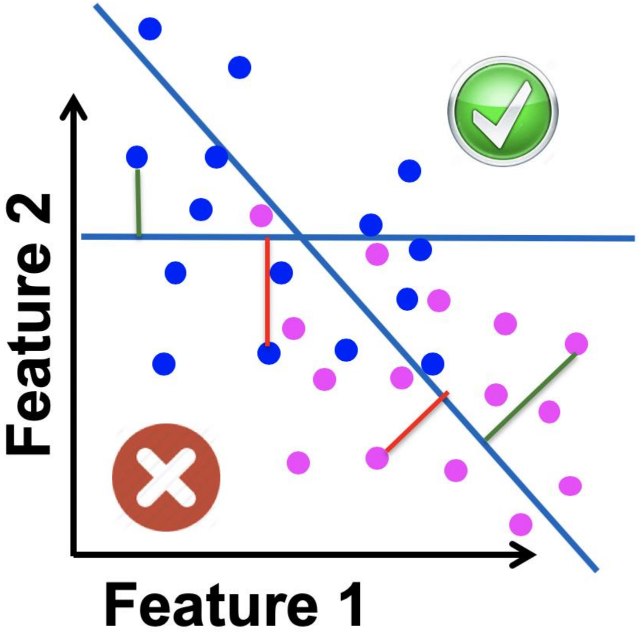
Artificial Intelligence and Statistics, pp. 962-970. 2017.

- To resolve the non-convex optimization issue:
    - Proposes the (alternative) measure of "decision boundary (un)fairness" for convex margin-based classifiers such as SVM.

An alternative for disparate impact

- The difference between the strength of acceptance and rejection across different demographic groups.

- The covariance between demographic groups and their signed distance from classifier’s decision boundary as the fairness measure



Decision-boundry fairness

$$cov\big(S, d_\theta(X)\big) = E[(S - \bar{S})d_\theta(X)] - E[(S - \bar{S})]E[d_\theta(X)]$$

$$\approx \frac{1}{n}\sum(S - \bar{S})d_\theta(X)$$

Considering the decision boundary at score zero: $\theta^\top X = 0$:

$$cov\big(S, d_\theta(X)\big) = \frac{1}{n}\sum_{i=1}^{n}(S_i - \bar{S})\theta^T X$$

Decision-boundry fairness:

$$\left|\frac{1}{n}\sum_{i=1}^{n}(S_i - \bar{S})\theta^T X\right| \leq \tau$$

Convex Optimization

- $\min L(\theta)$
- s.t.
  - $\frac{1}{n}\sum_{i=1}^{n}(S_i - \bar{S})\theta^T X \leq \tau$
  - $\frac{1}{n}\sum_{i=1}^{n}(\bar{S} - S_i)\theta^T X \geq -\tau$

Similar constraints can be applied for misclassification parity, false negative rate, and false positive rate parity

A reductions approach to fair classification

Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach

ICML 2018

1- How to handle different notions of fairness?

There is a cost associated with re-engineering the ML systems to satisfy fairness

→ This may be too much for many stakeholders

2- How to adopt the existing ML system?

* This paper can handle multiple sensitive attributes and multiple fairness measure

1- multiple fairness measures

- Define generic fairness constraints
- Each fairness constrains is defined as
- $\mu_j(\theta) = E\left[g_j\left(X, S, Y, f_\theta(X)\right) \middle| \varepsilon_j\right], \forall j \in$ demographic groups
  - ε_j does not depend on h → does not support measures based on sufficiency

- Example:
  - DP: $g_j\left(X, S, Y, f_\theta(X)\right) = f_\theta(x)$ and $\varepsilon_j = \{S = S_j\}$, $\varepsilon_* = true$
  - EO: $g_j\left(X, S, Y, f_\theta(X)\right) = f_\theta(x)$ and $\varepsilon_j = \left\{S = S_j, Y = y\right\}$, $\varepsilon_* = \{Y = y\}$
- Constraints:
  - $\mu_j(\theta) - \mu_*(\theta) \leq \tau$
  - $-\mu_j(\theta) + \mu_*(\theta) \leq \tau$

$$\Rightarrow \quad M\mu(\theta) \leq \tau$$

2- adopt the existing ML system

- Solution: build a wrapper around the existing learning system that ensures fairness
    - Key idea: reduce fair classification to a sequence of cost-sensitive classification problems, whose solutions yield a (randomized) classifier with the lowest (empirical) error subject to the desired constraints

→ The fairness component can seamlessly integrate to the system

- Find the classifier f that
  1. Minimizes the loss (classification error)
  2. Satisfies fairness constraints
- Iteratively call the black-box learner and apply reweighting and (possibly) relabeling the data
- It guarantees to find the most accurate fair classifier in not too many iterations (~5 in experiments)

Theorem: After $O(n^2 \log \#constraints)$ iterations, finds the classifier with probability $(1-\delta)$

- $\min_{\forall\theta} \; L(\theta) \quad$ **s.t.** $\quad M\mu(\theta) \le \tau$

- **Lagrangian dual form:** $L(\theta,\lambda) = L(\theta) + \lambda(M\mu(\theta) - \tau)$
- **Solve for Saddle point:**

$$\max_{\lambda} \boxed{\min_{\theta} L(\theta,\lambda)}$$

Existing ML system

Iterate while reweighting examples

Classification with fairness constraints: A meta-algorithm with provable guarantees

Elisa Celis, Lingxiao Huang, Vijay Keswani, and Nisheeth K. Vishnoi

FAT* 2019

- Proposes a meta-algorithm for a **general class of fairness** constraints with respect to **multiple** non-disjoint and **multi-valued** sensitive attributes

- Can handle non-convex linear fractional constraints, including predictive parity

Generalization of fairness functions: group performance function

- At a high-level, fairness requires equal "performance" of a classifier f for different demographic groups.

- For a classifier f, the group performance of group S_i is defined as

$$q_i(f) = P[\varepsilon | S_i, \varepsilon']$$

- Example:
  - Accuracy rate: $\varepsilon := (f = y), \varepsilon' := \emptyset$
  - False negative rate: $\varepsilon := (f = 0), \varepsilon' := (y = 1)$

The family of classifications with linear constraints

		$q_i(f)$		$Q_{\text{lin}}/Q_{\text{linf}}$
		$\mathcal{E}$	$\mathcal{E}'$	
fairness metrics	statistical	$f = 1$	$\emptyset$	Q_{lin}
	conditional statistical	$f = 1$	$X \in S$	Q_{lin}
	false positive	$f = 1$	$Y = 0$	Q_{lin}
	false negative	$f = 0$	$Y = 1$	Q_{lin}
	true positive	$f = 1$	$Y = 1$	Q_{lin}
	true negative	$f = 0$	$Y = 0$	Q_{lin}
	accuracy	$f = Y$	$\emptyset$	Q_{lin}
	false discovery	$Y = 0$	$f = 1$	Q_{linf}
	false omission	$Y = 1$	$f = 0$	Q_{linf}
	positive predictive	$Y = 1$	$f = 1$	Q_{linf}
	negative predictive	$Y = 0$	$f = 0$	Q_{linf}

ρ-Fair formulation

Nonconvex

- $\min_{\forall\theta} \; L(\theta)$

Loss term

- s.t.
  - $\rho_{q^{(i)}}(f_\theta) = \dfrac{\min q_j^{(i)}}{\max q_j^{(i)}} \geq \tau$

Fairness Constraint

*: $q^{(1)}$ … $q^{(m)}$ are the performance functions

Group-fair formulation

$$\min_{\forall\theta} \; L(\theta)$$

Loss term

s.t.

$$\ell_j^i \le q_j^{(i)}(f_\theta) \le u_j^i, \forall\, i \in [m], j \in [p]$$

Fairness Constraint

- Fairness constraints are linear → Convex

For any feasible classifier f of Group-Fair and any $i \in [m]$, f satisfies ρ-fair rule for:

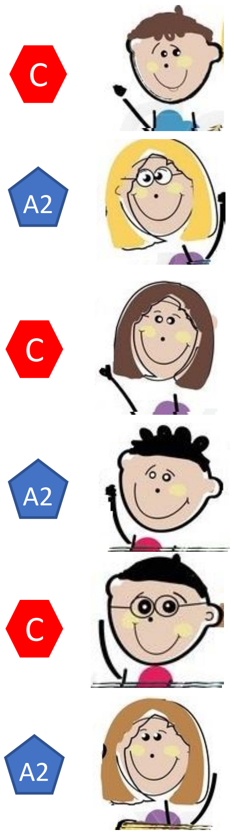
$$\rho = \frac{\min \ell_j^{(i)}}{\max u_j^{(i)}}$$

Ranking

Toy Example

A2 Ann Arbor

D Chicago



Suppose you own a real estate agency with two branches in Ann Arbor and Chicago.

You want to give bonus to

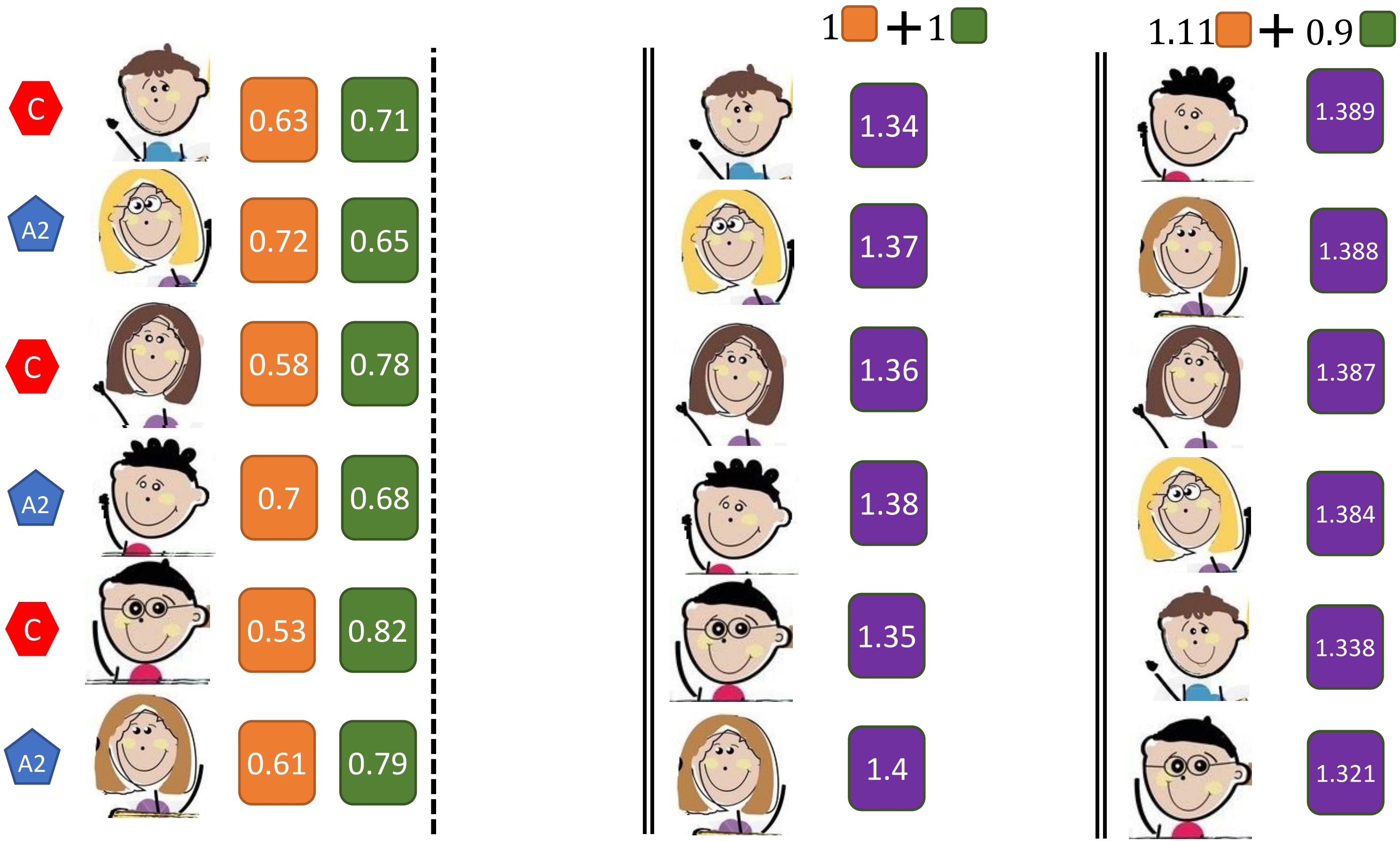
(1) Top-3 agents

To be fair, you want to make sure that each branch receives at least one promotion

Toy Example

Sale -- Normalized

Customer Satisfaction -- Normalized



$$\sum \theta_i x_i$$

$$\theta_1 \square + \theta_2 \square$$

Despite the potential impact of these weights, those are

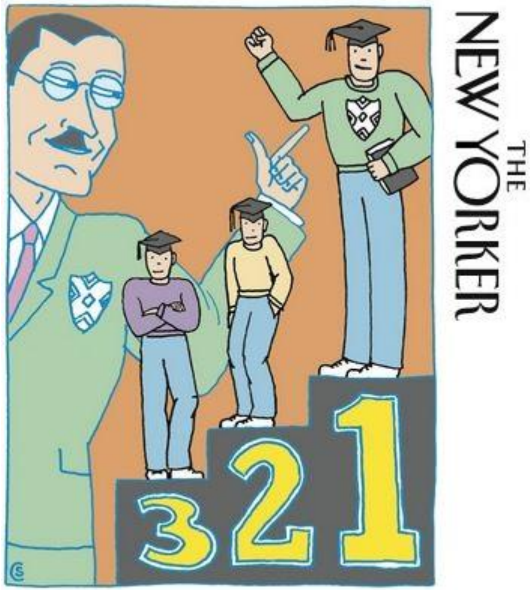
chosen in an ad-hoc manner!

THE ORDER OF THINGS

What college rankings really tell us.

By Malcolm Gladwell

- “It is easy to see why the U.S. News rankings are so popular. A single score allows us to judge between entities”
- “Rankings depend on what weights we give to what variables”
- “This idea of using the rankings as a benchmark, college presidents setting a goal of ‘We’re going to rise in the *U.S. News* ranking’ …”



Rankings depend on what weight we give to what variables.

Illustration by SEYMOUR CHWAST

Designing Fair Ranking Schemes

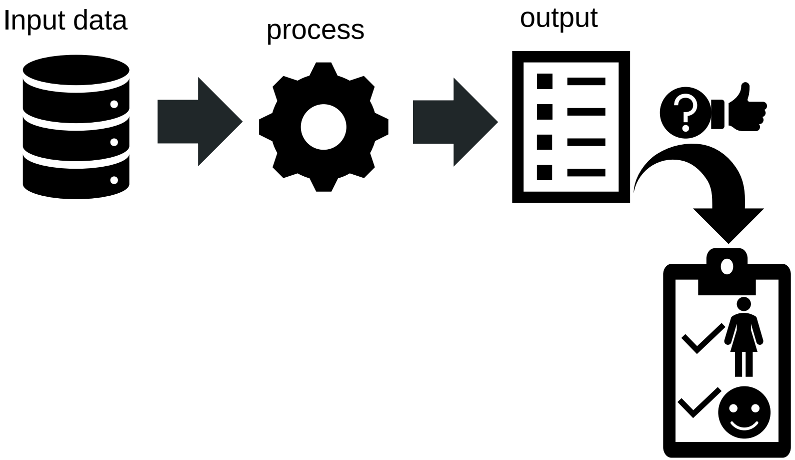
Abolfazl Asudeh, H. V. Jagadish, Julia Stoyanovich, and Gautam Das

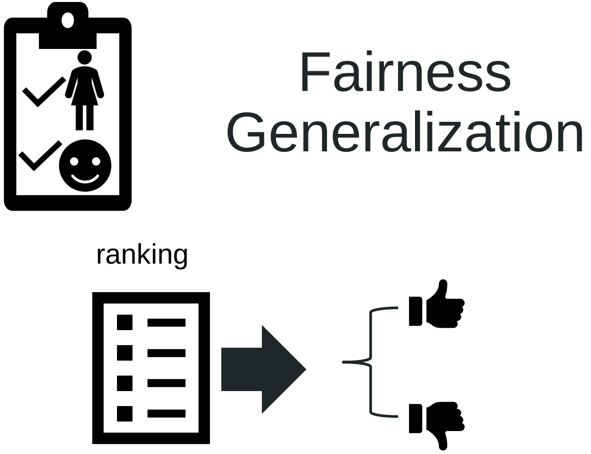
SIGMOD 2019

Fairness Model:

to support human values

- *Generate Fair outcomes*
- *Without Disparate Treatment*: ***explicit use of sensitive attributes*** to make decisions
    - not allowed in many jurisdictions

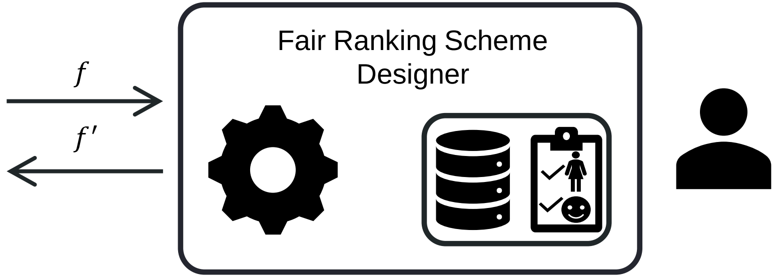




High level idea

- *Offline*: Preprocess the data and generate some indices
  - OK not to be super fast

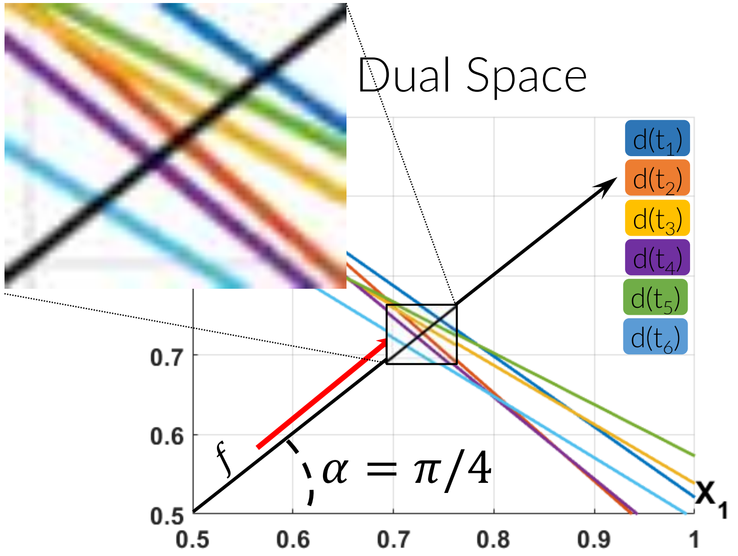
- *Online*: Answer user queries
  - Should be fast



2D Algorithm

Geometric interpretation

$\mathcal{D}$			f
id	x_1	x_2	$x_1 + x_2$
t_1	0.63	0.71	1.34
t_2	0.72	0.65	1.37
t_3	0.58	0.78	1.36
t_4	0.7	0.68	1.38
t_5	0.53	0.82	1.35
t_6	0.61	0.79	1.4



$d(t)$: $\sum t[i] \times x_i = 1$

2D:

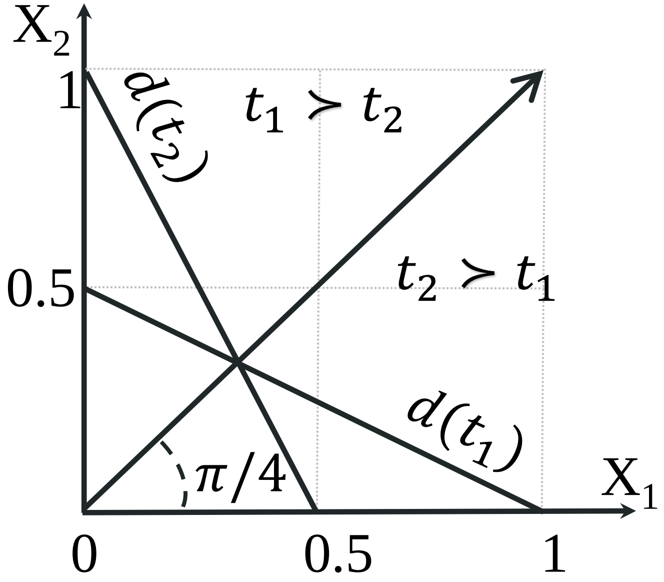
$d(t)$: $t[1]x_1 + t[2]x_2 = 1$

Ordering Exchange

- example

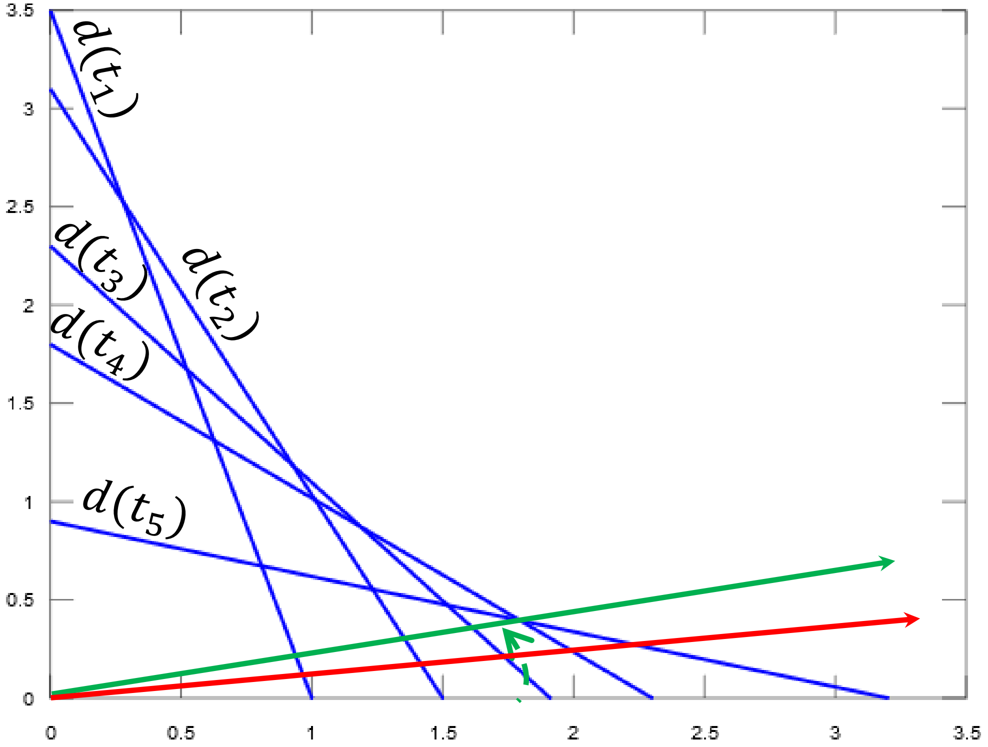
$t_1 < 1{,}2 >$

$t_2 < 2{,}1 >$



Ranking Regions

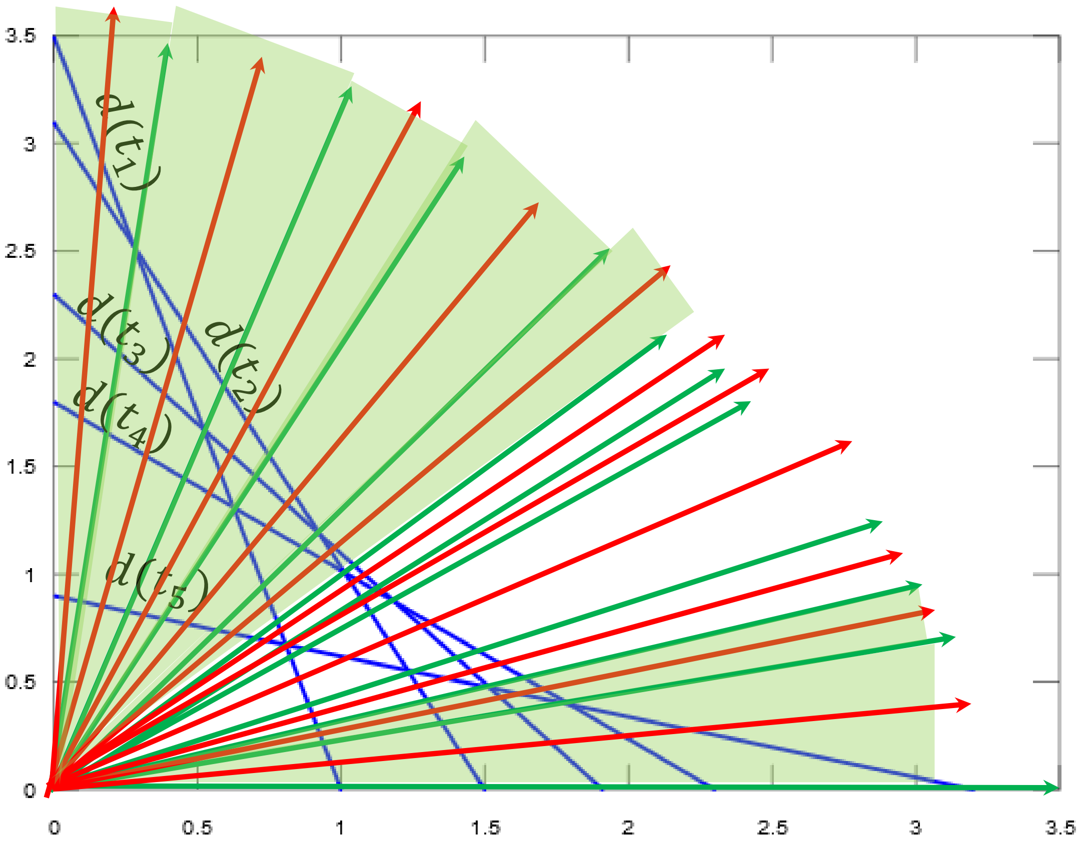
	x_1	x_2	location
t_1	3.5	1	A2
t_2	3.1	1.5	A2
t_3	2.3	1.91	C
t_4	1.8	2.3	C
t_5	0.9	3.2	A2



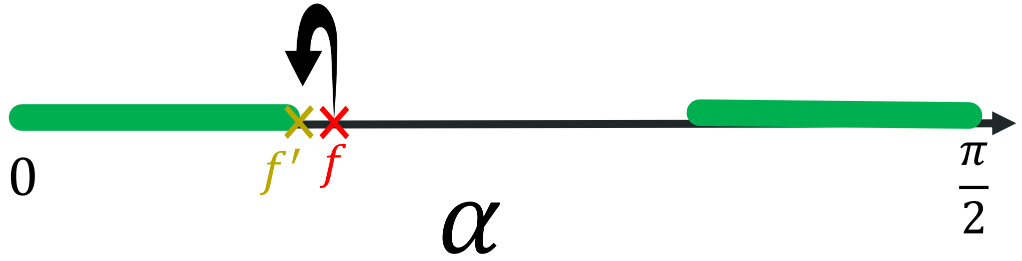
2D, offline:

	x_1	x_2	location
t_1	3.5	1	C
t_2	3.1	1.5	A2
t_3	2.3	1.91	C
t_4	1.8	2.3	A2
t_5	0.9	3.2	C

Fairness criterion:
at least one from each branch



2D: Online



- *Apply Binary Search!*

  fast: $O(\log n)$

* n: number of items

On obtaining stable rankings

Abolfazl Asudeh, H. V. Jagadish, Gerome Miklau, and Julia Stoyanovich

VLDB 2019

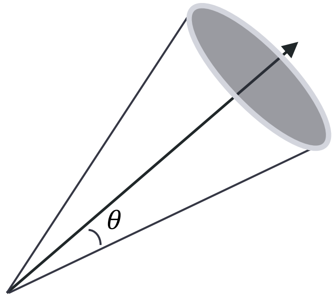
Stability: how robust the output is

- Small changes in weights change the output?
  - Decisions based on which are questionable (not fair)
  - Not Stable

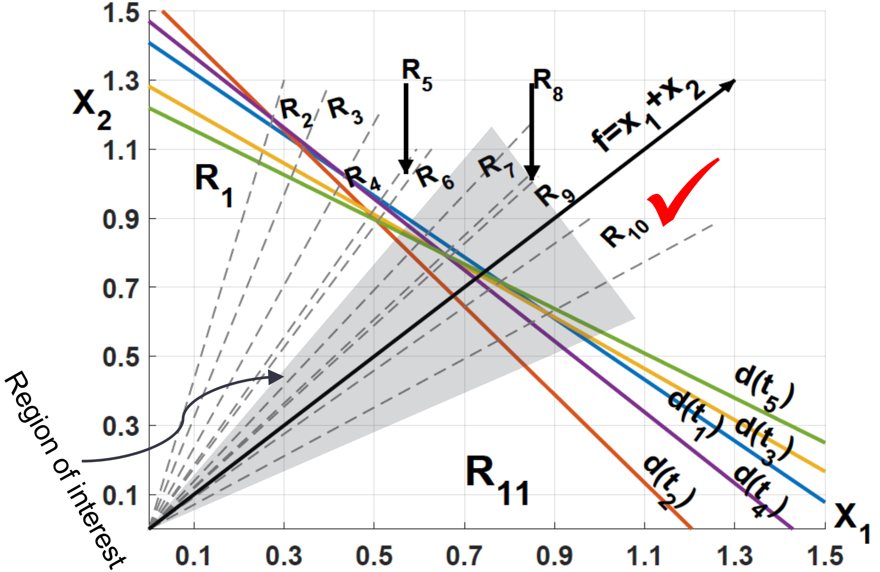
- Stability: The (volume) Ratio of functions that generate an output (ranking, top-k, or partial ranking)

Region of Interest

- The range of weights that are "acceptable" to the ranking designer
    - *A vector and angle distance: e.g.* at least 95% cosine similarity with a ref. vector



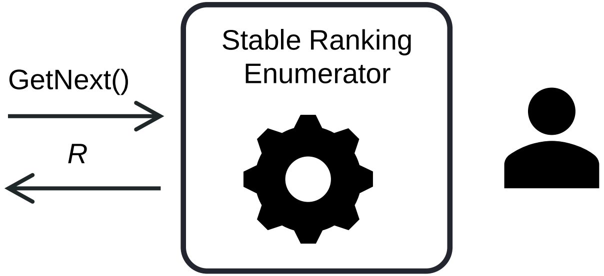
$\mathcal{D}$			f
id	x_1	x_2	$x_1 + x_2$
t_1	0.63	0.71	1.34
t_2	0.83	0.65	1.48
t_3	0.58	0.78	1.36
t_4	0.7	0.68	1.38
t_5	0.53	0.82	1.35



High level idea

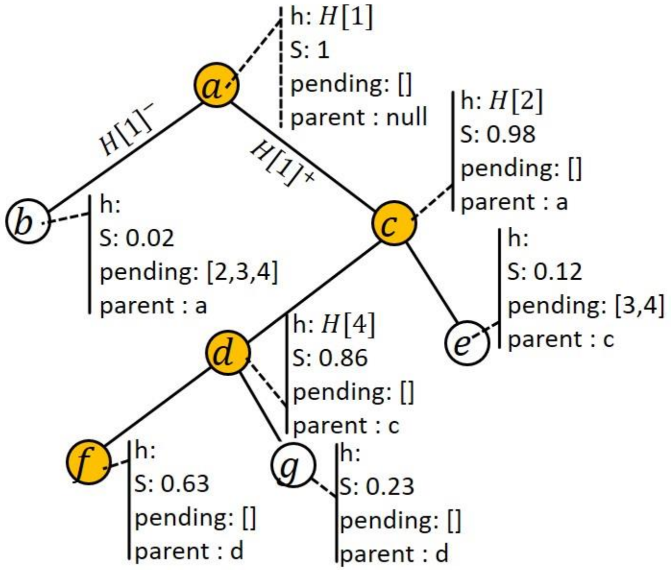
- *GetNext*: An iterative process that generate stable regions one after the other

- The user can keep enumerating stable rankings (or top-k), until he finds a satisfactory one



MD -- Threshold-based Algorithm

- Uses the arrangement tree
- In high-level:
    - Constructs the arrangement tree while only adds a postponing the process for the smaller regions

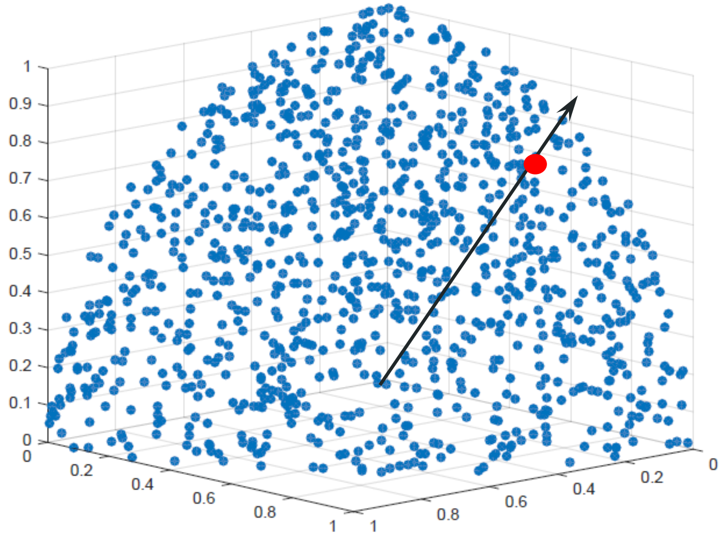


Randomized Get-Next

- A Monte-Carlo method that work based on repeated sampling and the central limit theorem

Unbiased sampling from the function space

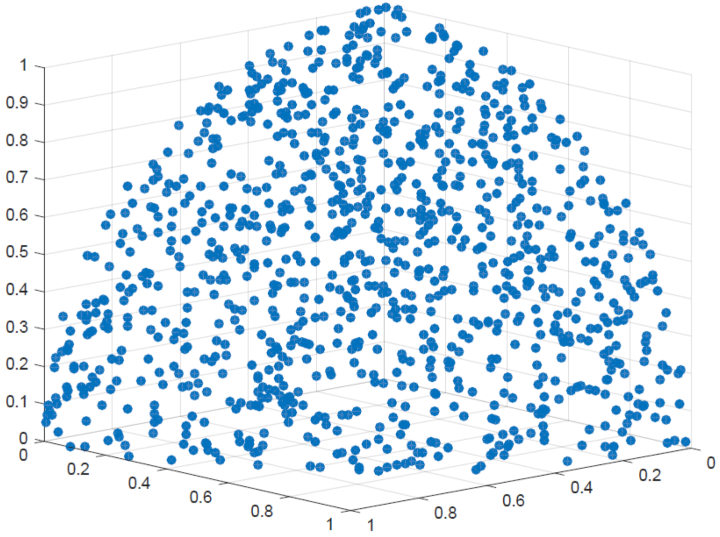
- **1-1 mapping b/w the functions (origin-starting rays) and the points on the** ***surface*** **of origin-centered unit** ***d-sphere*** **(hypersphere in $\mathbb{R}^d$)**



* d: number of attributes

Unbiased sampling from the function space

- Sampling the weights Uniformly? ✗
- Sampling the weights using the Normal distribution ✓



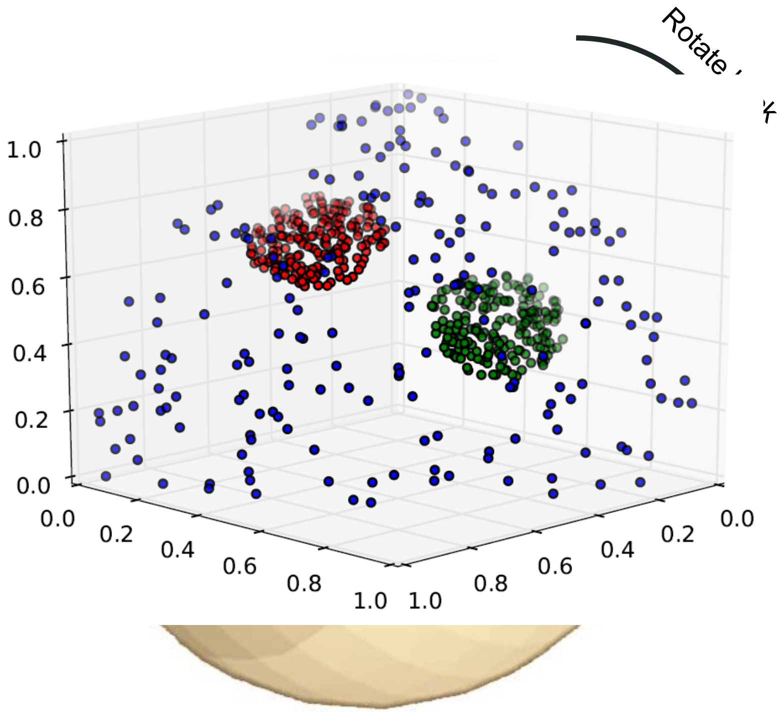
Asudeh, Abolfazl, and H. V. Jagadish. "Responsible Scoring Mechanisms Through Function Sampling." *arXiv preprint arXiv:1911.10073* (2019).

Sampling from a region of interest

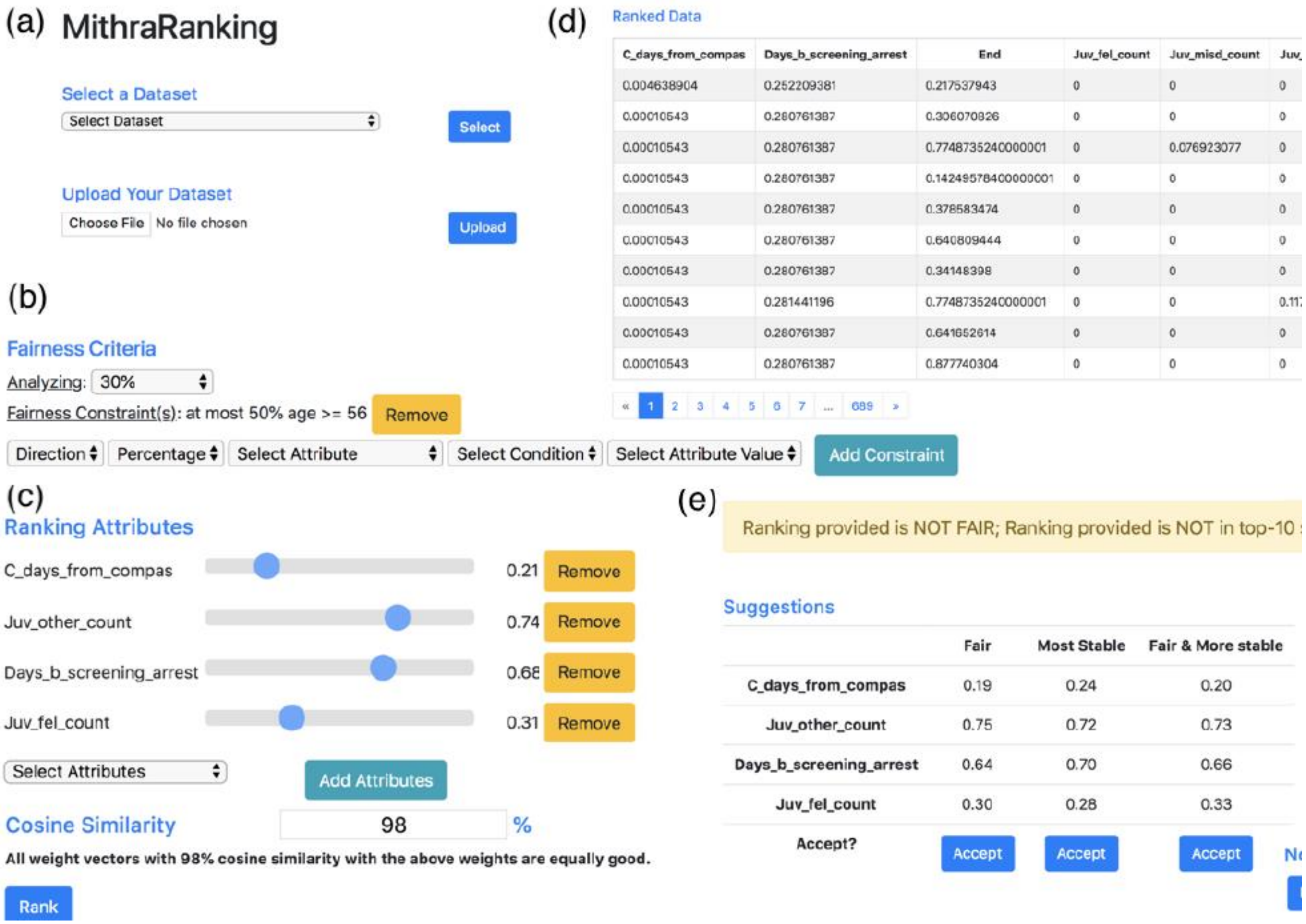
- Each Riemannian Piece is a (d-1)D Sphere (ring in 3D)
- We know how to sample from its surface!: *Normal distribution*

- High-level:

1. Select each "ring" randomly, proportional to its area
2. Select a "point" from the surface of ring (using the Normal dist.)
3. Rotate the space back



MithraRanking



[*] Yifan Guan, Abolfazl Asudeh, Pranav Mayuram, H. V. Jagadish, Julia Stoyanovich, Gerome Miklau, and Gautam Das. Mithraranking: A system for responsible ranking design. SIGMOD 2019.

Nutritional Labels

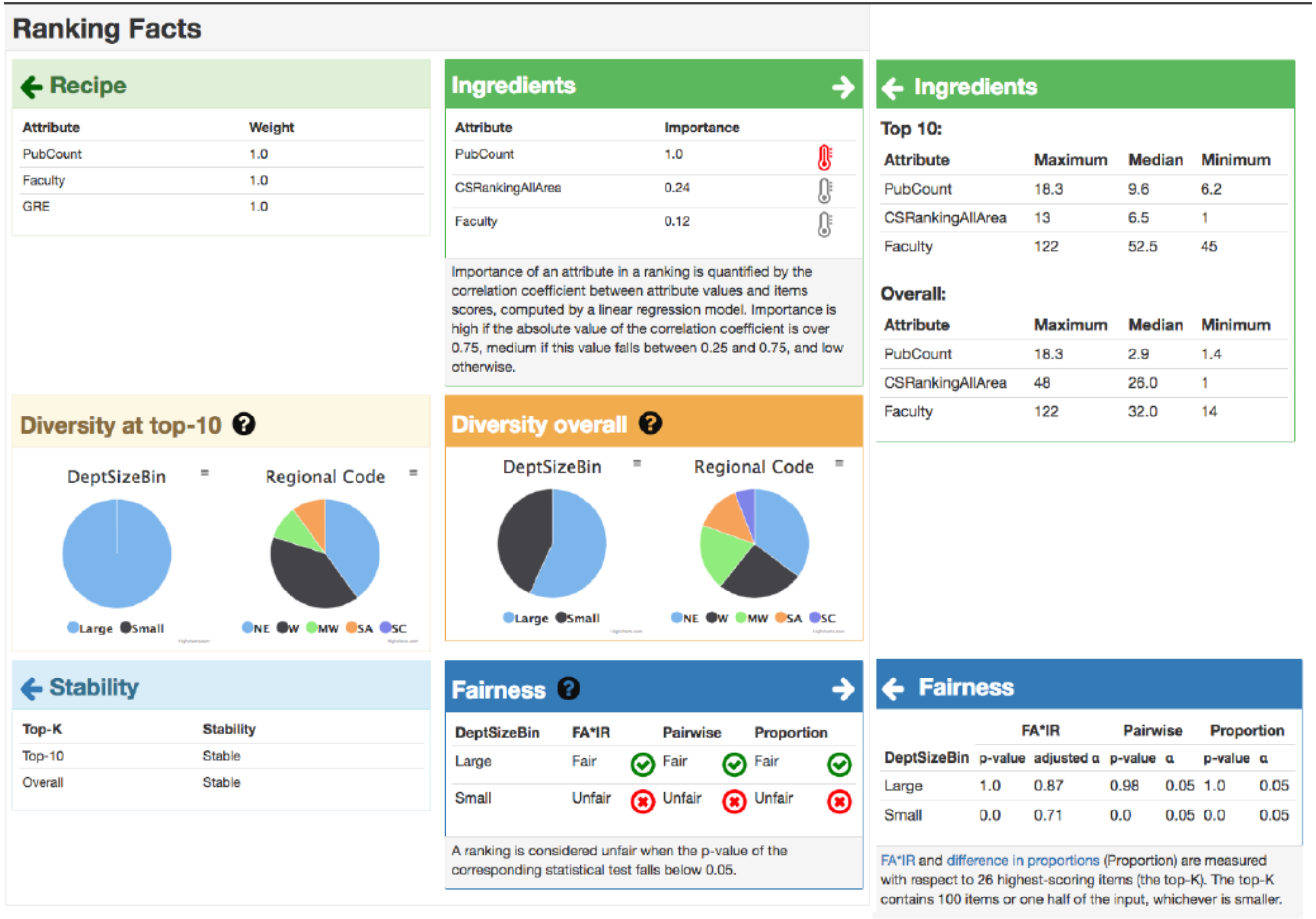
Nutritional labels for interpretability

- Interpretability is an essential ingredient of successful machine-assisted decision-making.
- This motivates creating tools that show deficiencies, biases, and unfairness in score-based evaluation.

- Drawing an analogy to the food industry, where simple, standard labels convey information about the ingredients and production processes:
  - a nutritional label is a set of automatically constructed visual widgets, each conveying standardized information about “fitness for use” of data or the evaluators

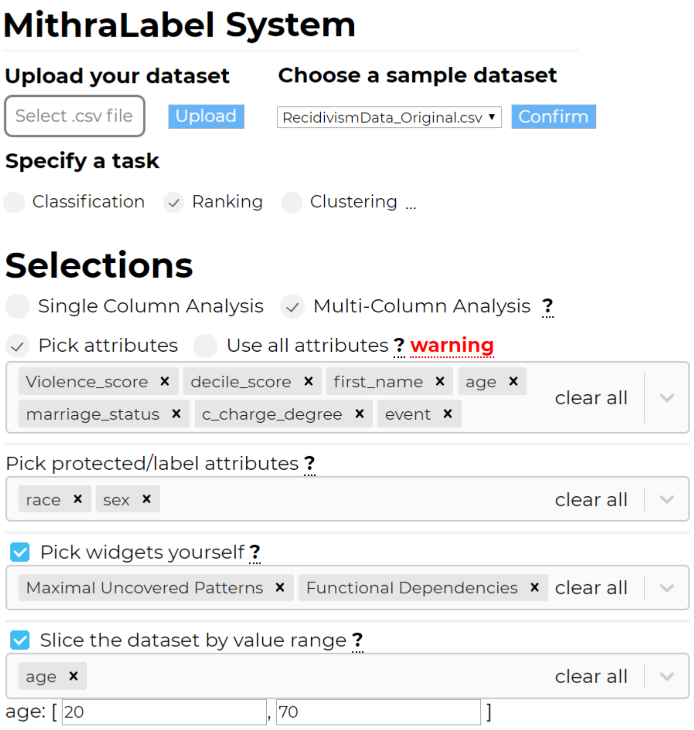
[*] Julia Stoyanovich, and Bill Howe. "Nutritional Labels for Data and Models." IEEE Data Eng. Bull. 42, no. 3 (2019): 13-23.

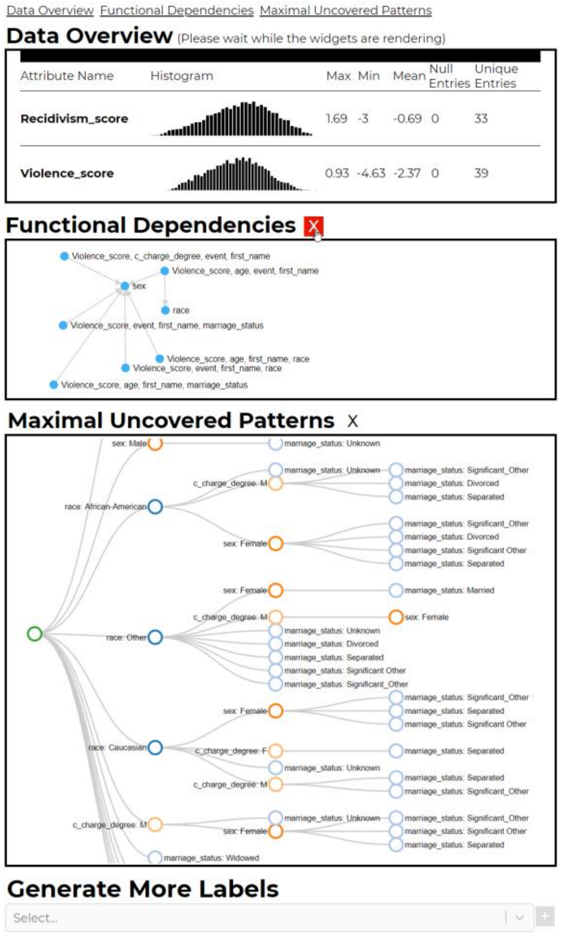
Ranking Facts: Nutritional Labels for Rankers



[*] Ke Yang, Julia Stoyanovich, A. Asudeh, Bill Howe, H. V. Jagadish, and G. Miklau.
A nutritional label for rankings.
In SIGMOD 2018.

MithraLabel: Flexible Data set Nutritional Labels





[*] C. Sun, A. Asudeh, H. V. Jagadish, B. Howe, and J. Stoyanovich. MithraLabel: Flexible dataset nutritional labels for responsible data science. In CIKM 2019