# Does Machine Bring in Extra Bias in Learning?
## Approximating Discrimination Within Models Quickly


Yijun Bian* [1] Yujie Luo* [2] Ping Xu [3]

[1]University of Copenhagen [2]National University of Singapore [3]The University of Texas Rio Grande Valley

UNIVERSITY OF COPENHAGEN


## Overview

We study the assessment of discrimination level of machine learning (ML) models when several sensitive attributes exist with multiple values, proposing

- a fairness metric (*Harmonic Fairness measure via Manifolds*, ***HFM***), by viewing instances with sensitive attributes as data points on certain manifolds
- two approximation algorithms (`ApproxDist` and `ExtendDist`) to quickly estimate the distance between sets—basis of *HFM*, accelerate the bias evaluation, and broaden its practical applicability

## Problem Statement and Motivation

Given a dataset $S$ composed of instances including sensitive attributes (SAs):

$$S=\{(\underbrace{\breve{\boldsymbol{x}}_i}_{\text{non-sensitive / unprotected}},\underbrace{\boldsymbol{a}_i}_{\text{sensitive / protected attributes}},y_i)\}_{i=1}^n\,,$$

where one instance is denoted by

$$\boldsymbol{x}=(\breve{\boldsymbol{x}},\boldsymbol{a})=[\overbrace{x_1,...,x_{n_x}}^{n_x\text{ is \# non-sensitive attributes}},\overbrace{a_1,...,a_{n_a}}^{n_a\text{ is \# sensitive attributes}}]^\mathsf{T}\,.$$

Inspired by individual fairness principle—similar treatment for similar individuals,

> *if viewing the instances (with the same SAs) as data points on certain manifolds, the manifold representing members from the marginalised group(s) is supposed to be as close as possible to that representing members from the privileged group.*

To measure the fairness in scenarios of one or more sensitive attributes, we get inspiration from 'the distance between sets' in mathematics.

## Proposed Fairness Metric: *HFM*

### Distance between sets for one bi-valued SA

For $n_a=1$ and $a_i\in\mathcal{A}_i=\{0,1\}$, the distance between two subsets—*the manifold(s) of marginalised group(s) and that of the privileged group*

$$\mathbf{D}.(S_1,\bar{S}_1)\triangleq\max\Big\{\max_{(\boldsymbol{x},y)\in S_1}\min_{(\boldsymbol{x}',y')\in\bar{S}_1}\mathbf{d}\big((\breve{\boldsymbol{x}},\ddot{y}),(\breve{\boldsymbol{x}}',\ddot{y}')\big),\ \max_{(\boldsymbol{x}',y')\in\bar{S}_1}\min_{(\boldsymbol{x},y)\in S_1}\mathbf{d}\big((\breve{\boldsymbol{x}},\ddot{y}),(\breve{\boldsymbol{x}}',\ddot{y}')\big)\Big\}\quad(1)$$

- $a_i=1$ means a member from the privileged group
- two disjoint subsets $S_1$ and $\bar{S}_1=S\setminus S_1=\{(\boldsymbol{x},y)\in S\,|\,a_i\neq1\}$
- a given specific distance metric $\mathbf{d}(\cdot,\cdot)$ (e.g., the standard Euclidean metric)
- a simplified notation $\ddot{y}$ that could be the true label $y$ or prediction $\hat{y}$
- (1) becomes $\mathbf{D}(S_1,\bar{S}_1)$ using $y$, and $\mathbf{D}_f(S_1,\bar{S}_1)$ when using $\hat{y}$ for classifiers

### Distance between sets for multi-valued SA(s)

When only one single sensitive attribute exists (i.e., $n_a=1$), let $\boldsymbol{a}=[a_i]^\mathsf{T}$, $a_i\in\mathcal{A}_i=\{1,2,...,n_{a_i}\}$, $n_{a_i}\geqslant3$, and $n_{a_i}\in\mathbb{Z}_+$. We extend (1) and introduce

i) *maximal distance measure for one sensitive attribute*

$$\mathbf{D}_{\cdot,\boldsymbol{a}}(S,a_i)\triangleq\max_{1\leqslant j\leqslant n_{a_i}}\Big\{\max_{(\boldsymbol{x},y)\in S_j}\overbrace{\min_{(\boldsymbol{x}',y')\in\bar{S}_j}\mathbf{d}\big((\breve{\boldsymbol{x}},\ddot{y}),(\breve{\boldsymbol{x}}',\ddot{y}')\big)}^{\text{to find the nearest point in }\bar{S}_j}\Big\}\quad(2)$$

ii) *average distance measure for one sensitive attribute*

$$\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S,a_i)\triangleq\tfrac{1}{n}\sum_{j=1}^{n_{a_i}}\sum_{(\boldsymbol{x},y)\in S_j}\min_{(\boldsymbol{x}',y')\in\bar{S}_j}\mathbf{d}\big((\breve{\boldsymbol{x}},\ddot{y}),(\breve{\boldsymbol{x}}',\ddot{y}')\big)\quad(3)$$

- a few disjoint subsets $S_j=\{(\boldsymbol{x},y)\in S\mid a_i=j\},\forall j\in\mathcal{A}_i$, and $\bar{S}_j=S\setminus S_j$
- in degenerate case $\mathbf{D}_{\cdot,\boldsymbol{a}}(S,a_i)=\mathbf{D}.(S_1,\bar{S}_1)$ when $\mathcal{A}_i=\{0,1\}$

When several sensitive attributes exist, that is, $\boldsymbol{a}=[a_1,...,a_{n_a}]^\mathsf{T}$, and each $a_i\in\mathcal{A}_i=\{1,2,..,n_{a_i}\}$, we have the generalised version

i) *maximal distance measure for sensitive attributes*

$$\mathbf{D}_{\cdot,\boldsymbol{a}}(S)\triangleq\max_{1\leqslant i\leqslant n_a}\mathbf{D}_{\cdot,\boldsymbol{a}}(S,a_i)\quad(4)$$

ii) *average distance measure for sensitive attributes*

$$\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S)\triangleq\tfrac{1}{n_a}\sum_{i=1}^{n_a}\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S,a_i)\quad(5)$$

- $n_{a_i}$ is the number of values for this sensitive attribute $a_i$ $(1\leqslant i\leqslant n_a)$

**Remark.** (1) It is easy to see that $\mathbf{D}_{\cdot,\boldsymbol{a}}(S)\geqslant\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S)$. (2) Both $\mathbf{D}_{\cdot,\boldsymbol{a}}(S,a_i)$ and $\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S,a_i)$ measure the fairness regarding the sensitive attribute $a_i$.

### Fairness metric in model assessment: *HFM*

$$\mathbf{df}_{\text{prev}}(f)=\mathbf{D}_{f,\boldsymbol{a}}(S)/\mathbf{D}_{\boldsymbol{a}}(S)-1\quad(6a)$$
$$\mathbf{df}(f)=\log\big(\mathbf{D}_{f,\boldsymbol{a}}(S)/\mathbf{D}_{\boldsymbol{a}}(S)\big)\quad(6b)$$
$$\mathbf{df}^{\text{avg}}(f)=\log\big(\mathbf{D}^{\text{avg}}_{f,\boldsymbol{a}}(S)/\mathbf{D}^{\text{avg}}_{\boldsymbol{a}}(S)\big)\quad(6c)$$

- $\mathbf{D}_{\boldsymbol{a}}(S)$, $\mathbf{D}^{\text{avg}}_{\boldsymbol{a}}(S)$ indicate the biases from the data
- $\mathbf{D}_{f,\boldsymbol{a}}(S)$, $\mathbf{D}^{\text{avg}}_{f,\boldsymbol{a}}(S)$ indicate the extra biases from the learning algorithm

## Proposed Approximation Algorithms

### Approximation of distances between sets for Euclidean spaces

To estimate the distance between data points inside $\mathcal{X}\times\mathcal{Y}$,

$$g(\boldsymbol{x},\ddot{y};\boldsymbol{w})=g(\breve{\boldsymbol{x}},\boldsymbol{a},\ddot{y};\boldsymbol{w})=[\ddot{y},x_1,...,x_{n_x}]^\mathsf{T}\boldsymbol{w}\,,\quad(7)$$

- a random projection $g:\mathcal{X}\times\mathcal{Y}\mapsto\mathbb{R}$
- a non-zero random vector $\boldsymbol{w}=[w_0,w_1,...,w_{n_x}]^\mathsf{T}$

> *the distance between similar data points tends to be closer than others after projecting them onto a general one-dimensional linear subspace*

After sorting all the projected data points on $\mathbb{R}$, it is likely that for one $(\boldsymbol{x},y)$ in $S_j$, the desired instance $\operatorname{argmin}_{(\boldsymbol{x}',y')\in\bar{S}_j}\mathbf{d}\big((\breve{\boldsymbol{x}},y),(\breve{\boldsymbol{x}}',y')\big)$ would be somewhere near it after the projection, and vice versa. Thus, searching for it could be accelerated by checking several adjacent instances rather than traversing the whole dataset.

### *ExtendDist* & *ApproxDist*

- Algorithm 3. *ExtendDist* — to estimate $\mathbf{D}_{\cdot,\boldsymbol{a}}(S)$ and $\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S)$

> For $j$ from 1 to $n_a$
> - $d^{(j)}_{\max},d^{(j)}_{\text{avg}}=$ `ApproxDist`$(\{(\breve{\boldsymbol{x}}_i,a_{i,j})\}_{i=1}^n,\{\ddot{y}_i\}_{i=1}^n;m_1,m_2)$
>
> Return $\max_{1\leqslant j\leqslant n_a}\{d^{(j)}_{\max}\mid j\in[n_a]\}$ and $\frac{1}{n_a}\sum_{j=1}^{n_a}d^{(j)}_{\text{avg}}$

- Algorithm 2. *ApproxDist* — to estimate $\mathbf{D}_{\cdot,\boldsymbol{a}}(S,a_i)$ and $\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S,a_i)$

> For $j$ from 1 to $m_1$
> - Take two orthogonal vectors $\boldsymbol{w}_0$ and $\boldsymbol{w}_1$ where each $\boldsymbol{w}_k\in[-1,+1]^{1+n_x}$ $(k=\{0,1\})$
> - For $k$ from 0 to 1, get $t^k_{\max},t^k_{\text{avg}}=$`AcceleDist`$(\{(\breve{\boldsymbol{x}}_i,a_i)\}_{i=1}^n,\{\ddot{y}_i\}_{i=1}^n,\boldsymbol{w}_k;m_2)$
> - $d^j_{\max}=\min\{t^k_{\max}\mid k\in\{0,1\}\}=\min\{t^0_{\max},t^1_{\max}\}$
> - $d^j_{\text{avg}}=\min\{t^k_{\text{avg}}\mid k\in\{0,1\}\}=\min\{t^0_{\text{avg}},t^1_{\text{avg}}\}$
>
> Return $\min\{d^j_{\max}\mid j\in[m_1]\}$ and $\frac{1}{n}\min\{d^j_{\text{avg}}\mid j\in[m_1]\}$

- Algorithm 1. *AcceleDist* — to estimate $\mathbf{D}_{\cdot,\boldsymbol{a}}(S,a_i)$ and $n\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S,a_i)$

> Project data points onto a 1-dim space and obtain $\{g(\boldsymbol{x}_i,\ddot{y}_i;\boldsymbol{w})\}_{i=1}^n$
> Sort original data points using $g(\cdot,\cdot;\boldsymbol{w})$ in ascending order
> For $i$ from 1 to $n$
> - Set the anchor data point $(\boldsymbol{x}_i,\ddot{y}_i)$ in this round
>   `// If` $a_i=j$ `(marked for clarity), to approximate` $\min_{(\boldsymbol{x}',y')\in\bar{S}_j}\mathbf{d}(\text{anchor},(\breve{\boldsymbol{x}}',\ddot{y}'))$
> - Compute distances for *at most $m_2$ nearby data points* that meets $a\neq a_i$, $g\leqslant g_i$
> - Find the minimum among them, recorded as $d^s_{\min}$
> - Compute distances for *at most $m_2$ nearby data points* that meets $a\neq a_i$, $g\geqslant g_i$
> - Find the minimum among them, recorded as $d^r_{\min}$
> - $d^{(i)}_{\min}=\min\{d^s_{\min},d^r_{\min}\}$
>
> Return $\max\{d^{(i)}_{\min}\mid i\in[n]\}$ and $\sum_{i=1}^n d^{(i)}_{\min}$

- High computational complexity ($\mathcal{O}(n^2)$) of directly calculating (2) and (3)
- Reduced computational complexity ($\mathcal{O}(n\log n)$) of approximation algorithms

## Empirical Results

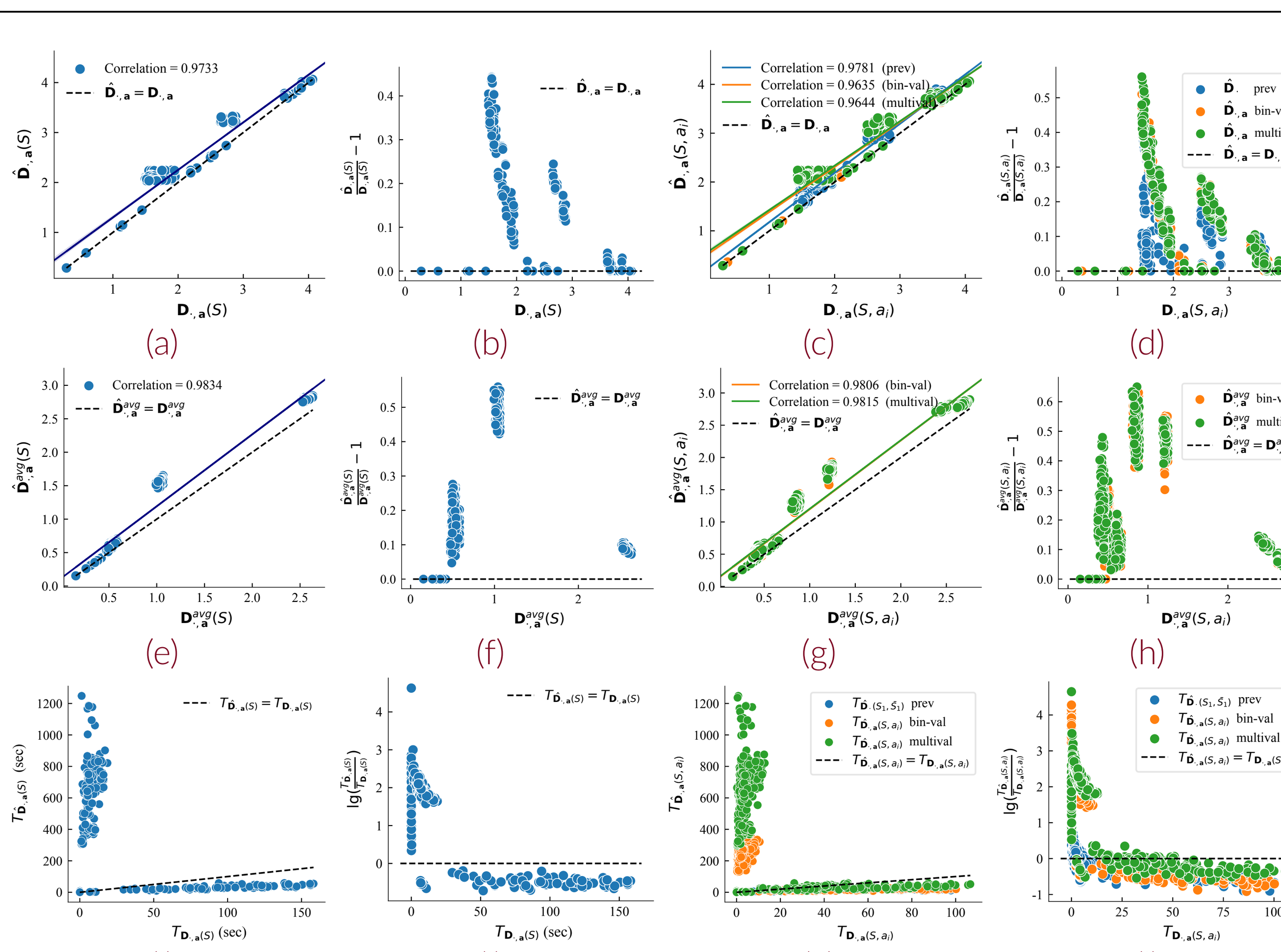


Figure 1. Comparison of approximated distances with precise values of definitions. (a–b), (c–d), (e–f), and (g–h) Scatter plots for comparison between approximated and precise values of $\mathbf{D}_{\cdot,\boldsymbol{a}}(S)$, $\mathbf{D}_{\cdot,\boldsymbol{a}}(S,a_i)$, $\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S)$, and $\mathbf{D}^{\text{avg}}_{\cdot,\boldsymbol{a}}(S,a_i)$, respectively; (i–j) and (k–l) Time cost comparison between approximation algorithms (*ExtendDist* and *ApproxDist*) and direct computation.

SCAN ME

*Looking for more details?*



yibi@di.ku.dk