Aequitas:

Automated Fairness Testing of Machine Learning Datasets

Yemi Shin, Yunping Wang, Michael Worrell, Juanito Zhang Yang

Table of contents

1
Introduction
Machine Learning &
Machine Learning Fairness

2
Literature Review
How Aequitas Works
Implementation

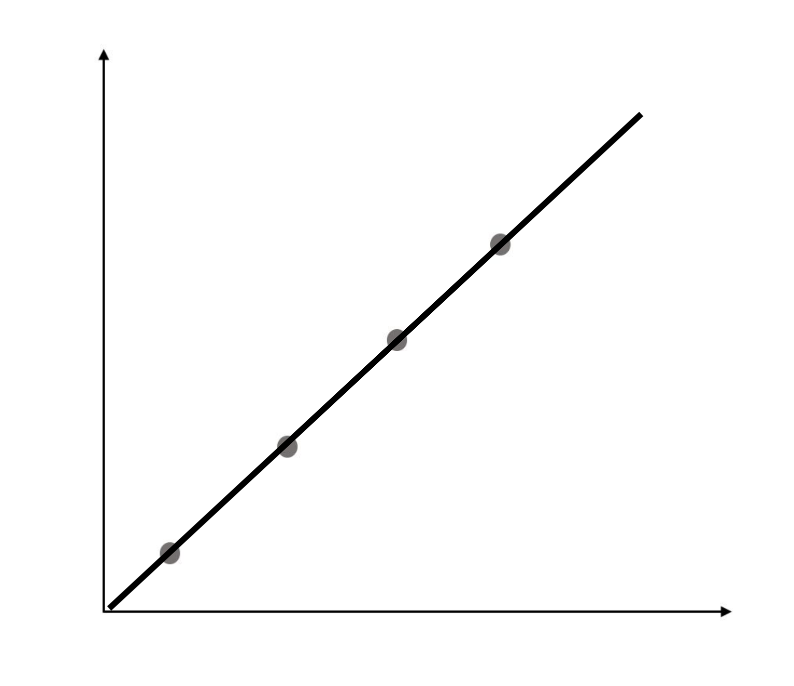
3
Our Work
Usability Improvements

4
Conclusion
Where to go from here

1
Introduction

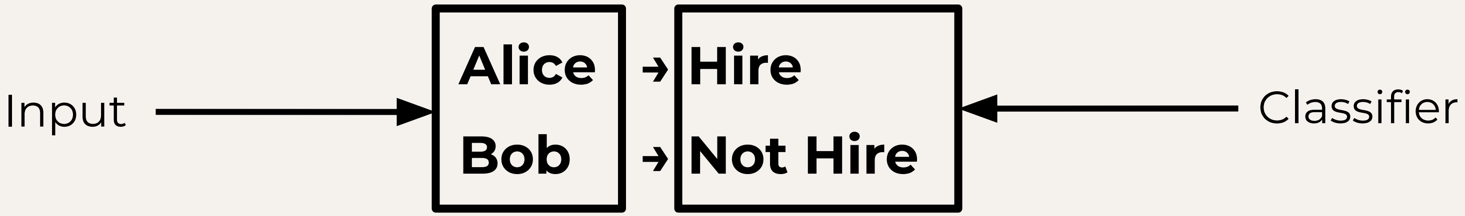
Machine learning is the process of finding functions from real-world data

- Training a machine learning model is the process of finding a best fit function.



Machine learning is the process of finding functions from real-world data

Hire	Not Hire
People whose name start with A	People whose does not start with A



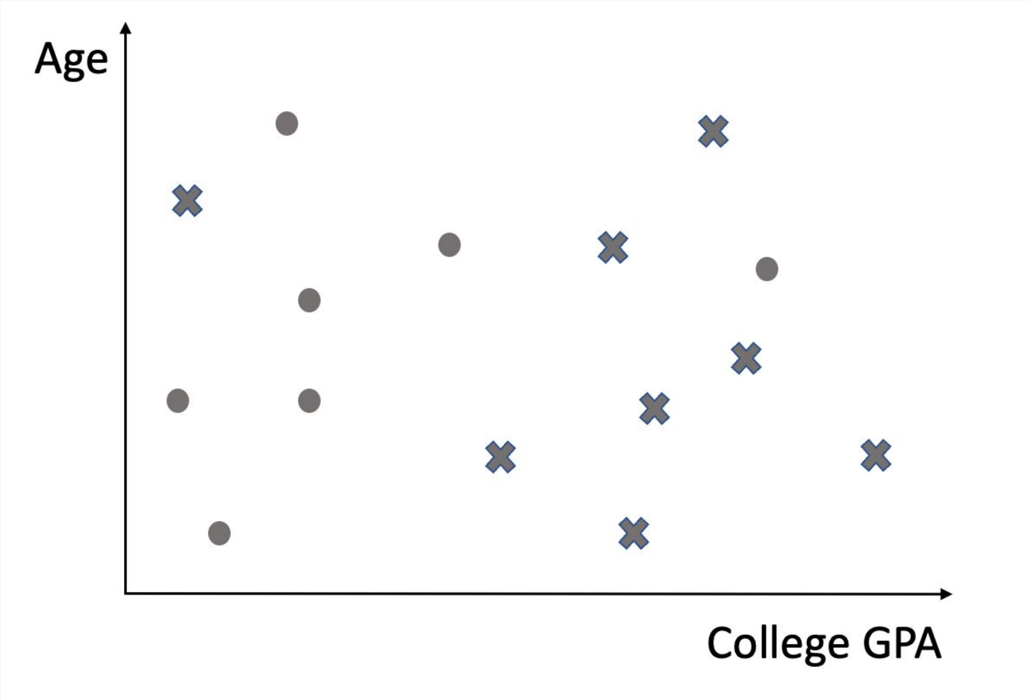
An input to a machine learning model can be treated as a vector

Name	Age	Major	Nationality	Education	...
Alice	34	CS	US	College	...

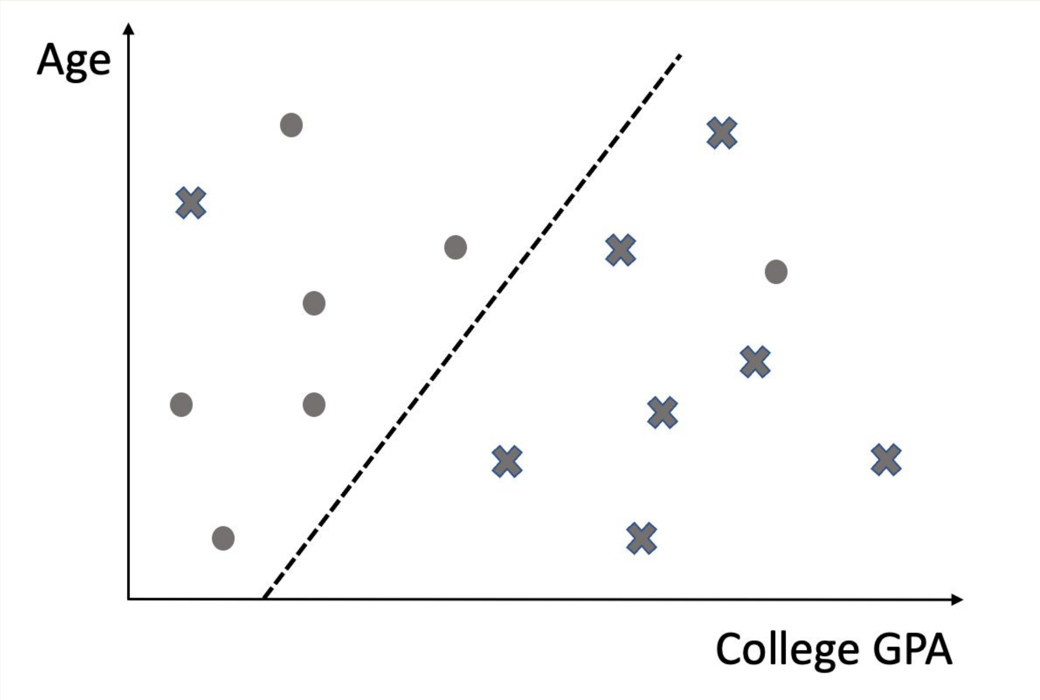
{“Alice”, “34”, “CS”, “US”, “College”}

Features

Decision boundary is another expression of machine learning model.



Decision boundary is another expression of machine learning model.



Introduction to Machine Learning Fairness

What is Machine Learning Fairness?

Importance of Machine Learning Fairness

Machine learning fairness is becoming increasingly important.

The **ACM Conference on Fairness, Accountability, and Transparency** is hosted annually in different parts of the globe.

FAT*2020
Barcelona, Spain

FAccT 2021

FAccT
서울
2022

Challenges of Machine Learning Fairness

How does one define **fairness**? How does one define **unfairness**?

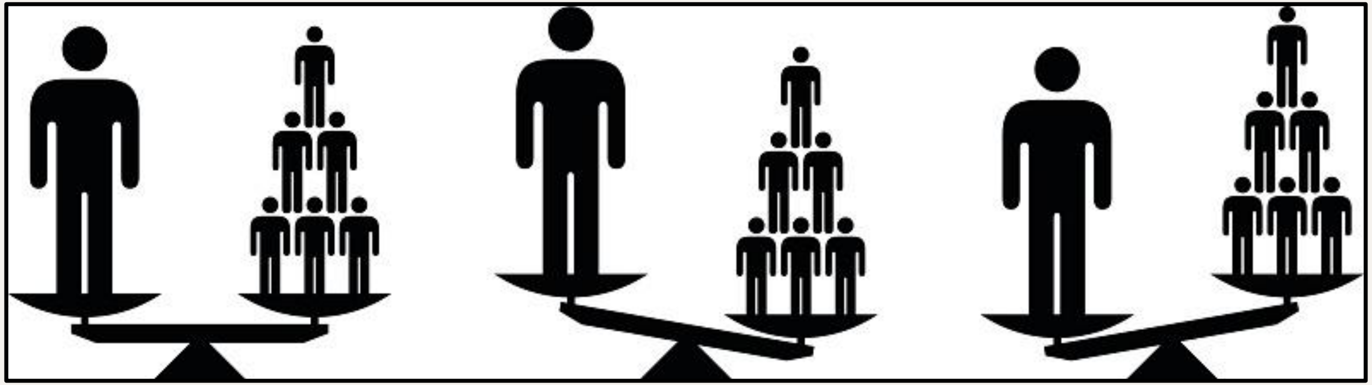
What **algorithms** does one use to identify and resolve unfairness?

How does one tradeoff between **fairness** and **accuracy**?

Defining Fairness in Machine Learning

There are different ways machine learning models can define fairness.

Different ways of thinking about fairness have different priorities.



Defining Fairness in Machine Learning

Aequitas runs using **individual fairness**.

Defining Fairness in Machine Learning

Aequitas runs using **individual fairness**.

	Sensitive Feature				Prediction
College	Female	NYC	25	2	
College	Non Binary	NYC	25	2	
Education	Gender	City	Age	# Years of Experience	Hired

Defining Fairness in Machine Learning

Aequitas runs using **individual fairness**.

	Sensitive Feature				Prediction
College	Female	NYC	25	2	
College	Non Binary	NYC	25	2	
Education	**Gender**	City	Age	# Years of Experience	Hired

Defining Fairness in Machine Learning

Aequitas runs using **individual fairness**.

	Sensitive Feature				Prediction
College	Female	NYC	25	2	Yes
College	Non Binary	NYC	25	2	Yes
Education	**Gender**	City	Age	# Years of Experience	Hired

Satisfies Individual Fairness!

Defining Fairness in Machine Learning

Aequitas runs using **individual fairness**.

Sensitive Feature | | | | | Prediction

Education	**Gender**	City	Age	# Years of Experience	Hired
College	Female	NYC	25	2	No
College	Non Binary	NYC	25	2	No

Satisfies Individual Fairness!

Algorithms in Machine Learning Fairness

Different strategies to deal with fairness that rely on three different categories of algorithms.

Pre-processing Algorithms – address biases in the model **data**.

Processing Algorithms – address biases during model **training**.

Post-processing Algorithms – address biases in model **output**.

Algorithms in Machine Learning Fairness

Aequitas is a **pre-processing** algorithm.

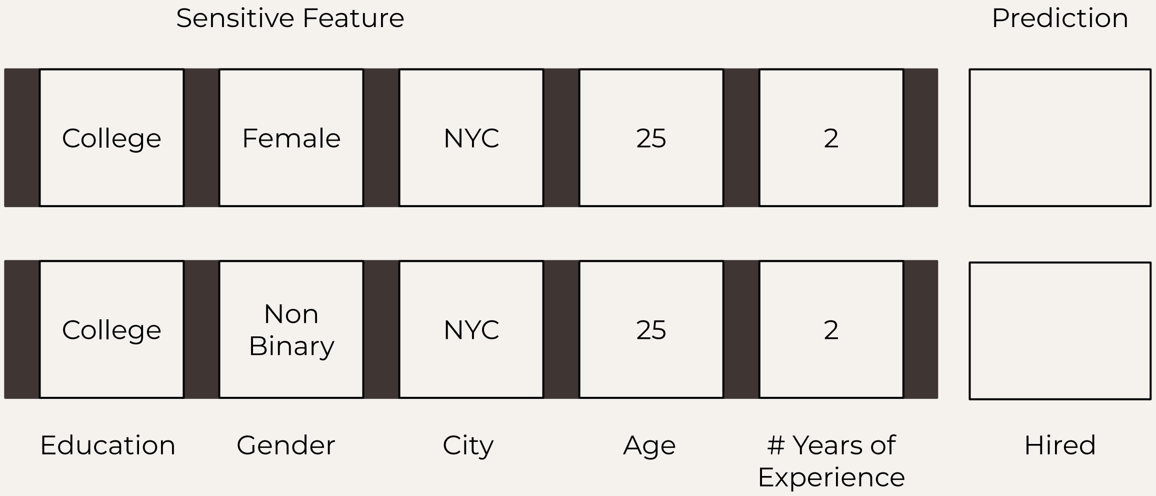
Discriminatory inputs identified using **counterfactual fairness**.

Algorithms in Machine Learning Fairness

Aequitas identifies discriminatory inputs using **counterfactual fairness**.

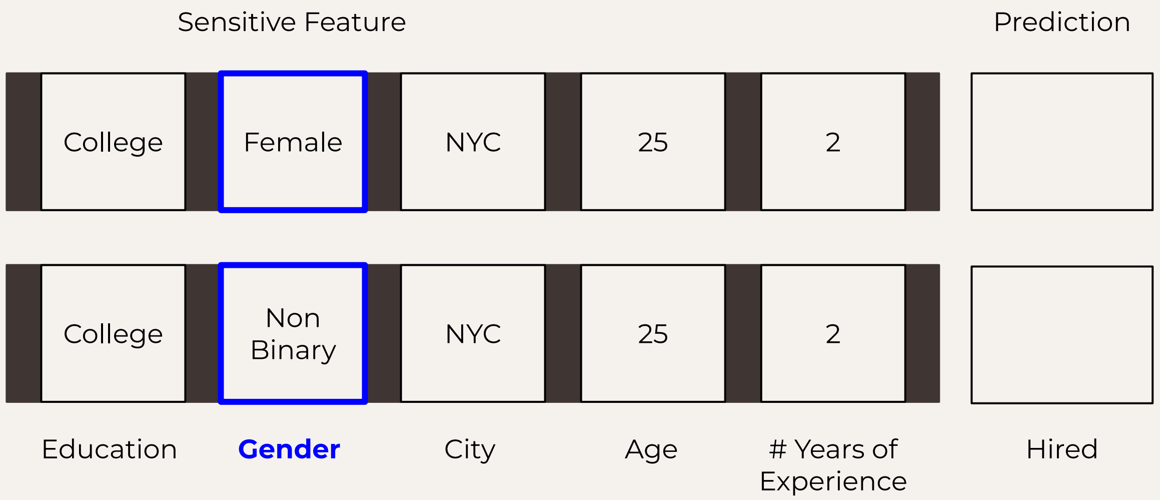
Algorithms in Machine Learning Fairness

Aequitas identifies discriminatory inputs using **counterfactual fairness**.



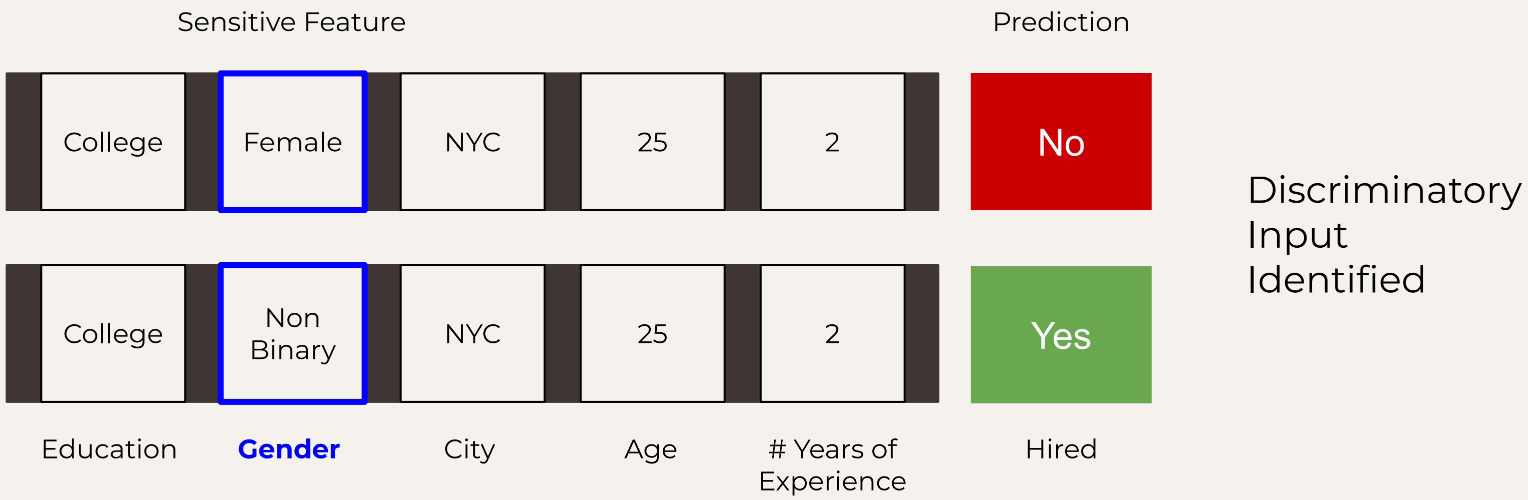
Algorithms in Machine Learning Fairness

Aequitas identifies discriminatory inputs using **counterfactual fairness**.



Algorithms in Machine Learning Fairness

Aequitas identifies discriminatory inputs using **counterfactual fairness**.



Algorithms in Machine Learning Fairness

Aequitas identifies discriminatory inputs using **counterfactual fairness**.

	Sensitive Feature				Prediction
College	Female	NYC	25	2	Yes
College	Non Binary	NYC	25	2	No
Education	**Gender**	City	Age	# Years of Experience	Hired

Discriminatory Input Identified

Fairness vs Accuracy in Machine Learning

When dealing with **fairness**, one also needs to consider how one affects the **accuracy** of a given model.

Solutions that reduce unfairness oftentimes sacrifice accuracy.

Can be addressed using **accuracy constraints** or **fairness constraints**.

Fairness vs Accuracy in Machine Learning

Accuracy constraints meet accuracy standards at the cost of fairness.

Fairness constraints meet fairness standards at the cost of accuracy.

Fairness vs Accuracy in Machine Learning

Aequitas uses a **fairness constraint**

Difference between input classification cannot exceed some numerical threshold $\boldsymbol{\gamma}$.

In the case of Aequitas, $\boldsymbol{\gamma = 0}$.

In Summary: Aequitas and Fairness

Aequitas is a machine learning fairness algorithm that looks for biased inputs using outputs from a previously trained model.

It utilizes counterfactual fairness to check whether individual fairness is being satisfied for random entries in the input domain.

2

Literature Review

Aequitas Theoretical Background

- Machine learning robustness.
- Introducing Aequitas:
  - Goals
  - Parameters
  - Key definitions.
- Retraining strategies.
- Overview of algorithm steps.

Robustness

- The output of machine learning classifiers is **not dramatically affected** by small changes to its inputs.

- Which means that inputs in **"the neighborhood"** of a discriminatory input **will have similar behavior**.

Aequitas

Automated Directed Fairness Testing

Sakshi Udeshi
Singapore Univ. of Tech. and Design
Singapore
sakshi_udeshi@mymail.sutd.edu.sg

Pryanshu Arora
BITS Pilani
India
pryanshu23@gmail.com

Sudipta Chattopadhyay
Singapore Univ. of Tech. and Design
Singapore
sudipta_chattopadhyay@sutd.edu.sg

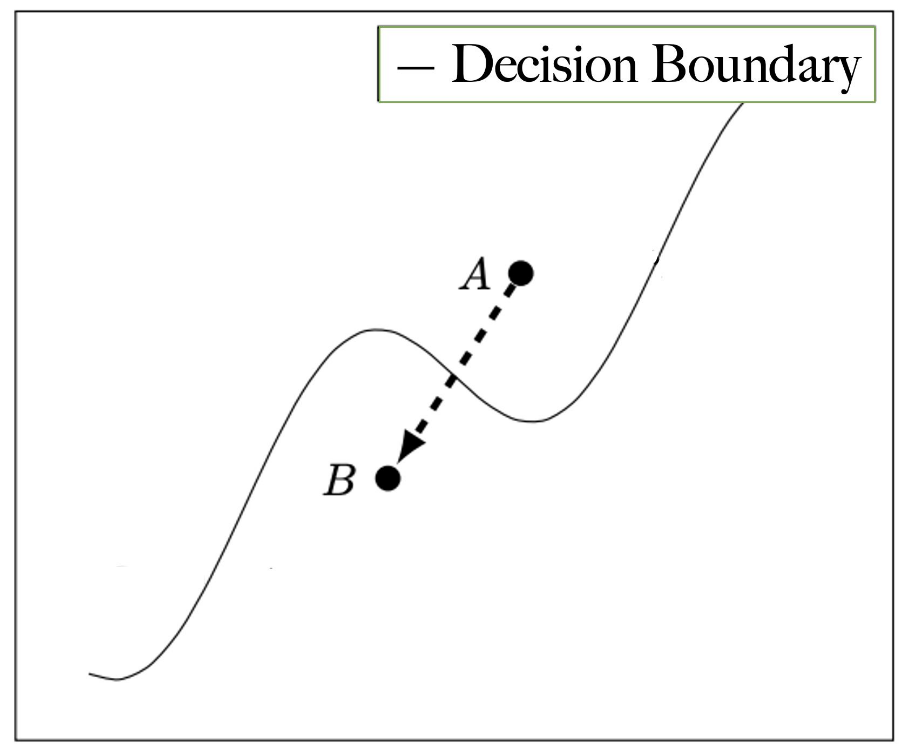
Aequitas: motivation

- Machine learning classifiers and their training datasets contain unintended biases.
- We want to design techniques to geneterate sets of **discriminatory inputs**.
- With this new dataset, we want to **iteratively retrain** our original classifier.

Aequitas: definitions

- ***Parameters:*** a machine learning classifier, a training dataset, and a set of sensitive features.
- ***Input space or input domain:*** the domain of the classifier, as a function.
- ***Perturbation:*** a perturbation of an input of the classifier is the same input, where one of features has been changed.

Aequitas: discriminatory inputs



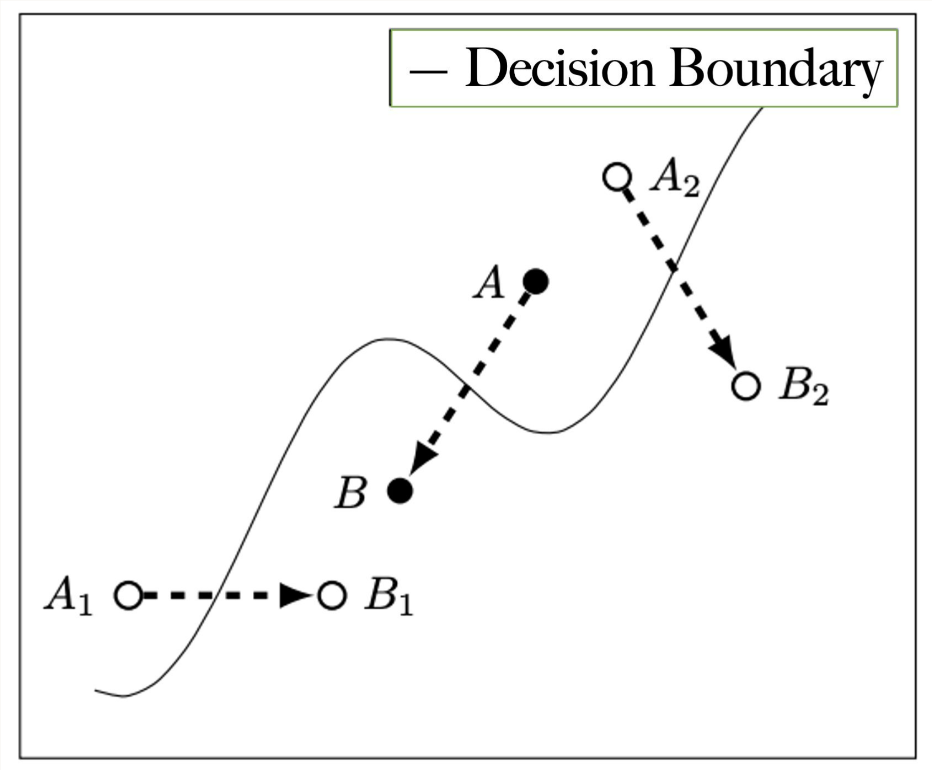
- A and B are inputs to the classifier,
- A and B have the **same characteristics except for their genders**,
- The classifier treats A and B differently.

Aequitas: discriminatory inputs

- Given a classifier and a set of **sensitive features**,

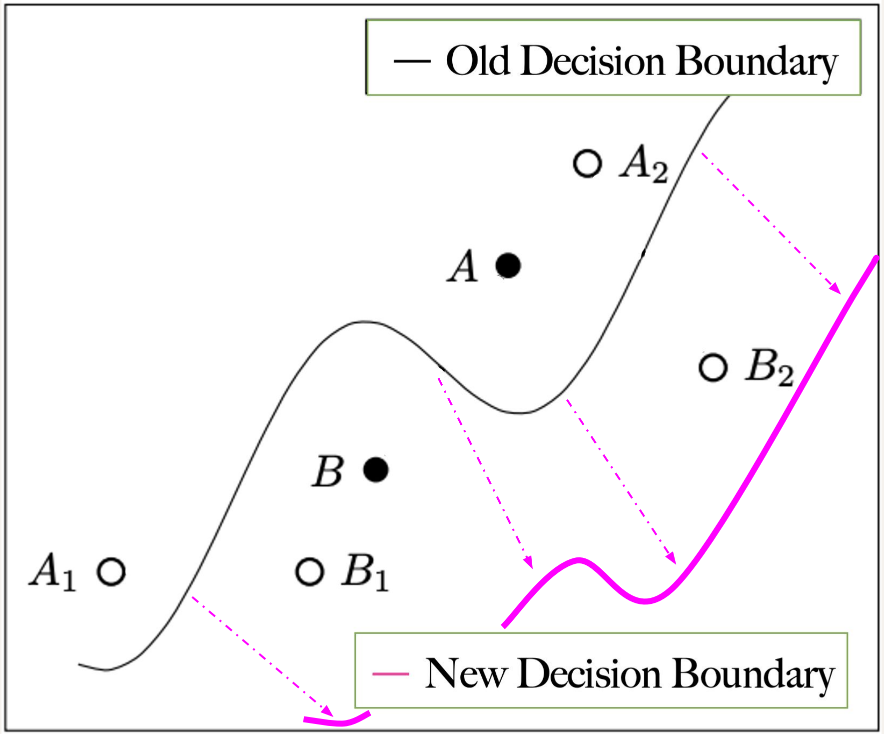
- An **input** $\boldsymbol{I}$ in the domains space is **discriminatory** if:
  - There is another input $\boldsymbol{I'}$ such that at **least one of the sensitive features is different** and all non-sensitive features are the same and,
  - The classification of $\boldsymbol{I}$ and $\boldsymbol{I'}$ are different.

Aequitas: robustness



- In the **neighborhood** of A and B,
- We may find **other inputs that are also discriminatory**, by robustness.

Retraining strategies



- By adding all of the **discriminatory inputs with the same label** to the dataset, we hope to shift the decision boundary,
- Now the inputs are treated equally.

Aequitas: overview

- ***Global search:*** Sample **uniformly at random** from the **input space** to find discriminatory inputs.

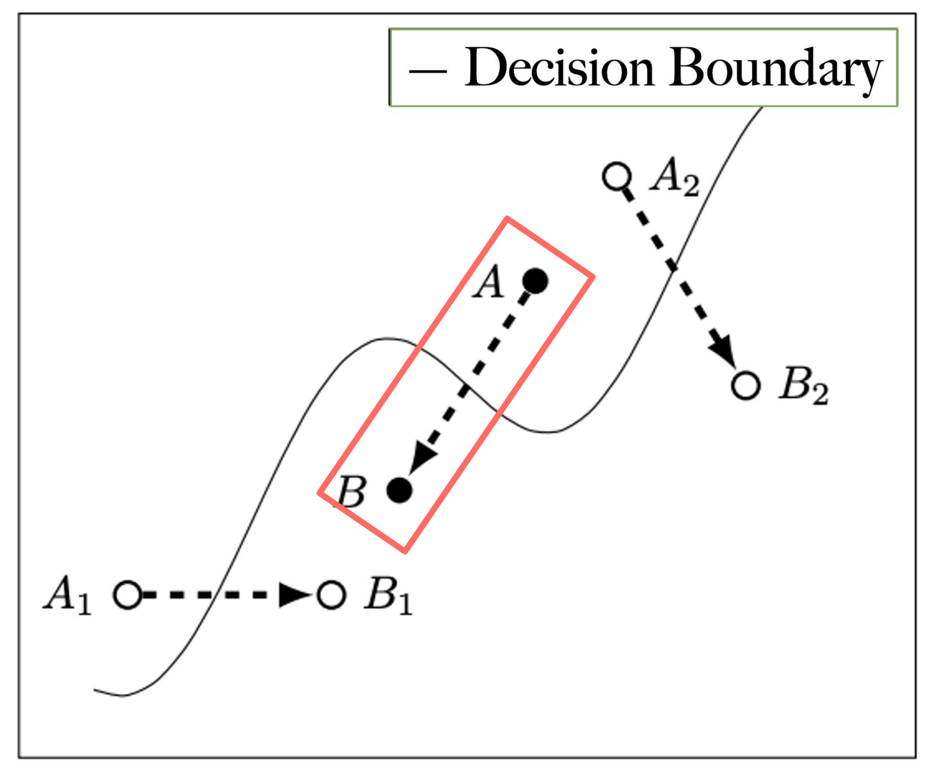
- ***Local search:*** Look in the **neighborhood** of each of the inputs of the preview step to **find more discriminatory** inputs, thanks to robustness.

- ***Retraining:*** Combine the original dataset and the inputs from both steps to retrain the classifier.

Implementation

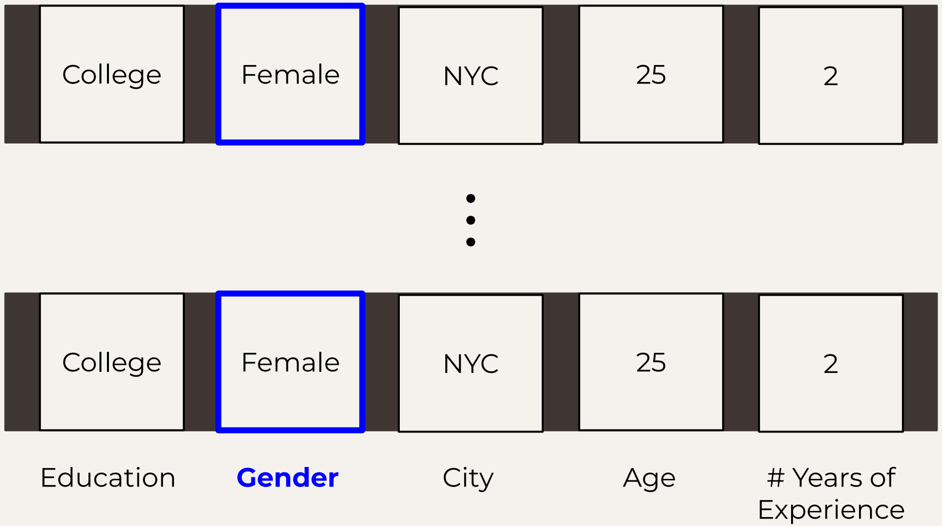
How do we fairly predict who gets hired?

Global Search



How do we evaluate fairness?

1. Clone the input.

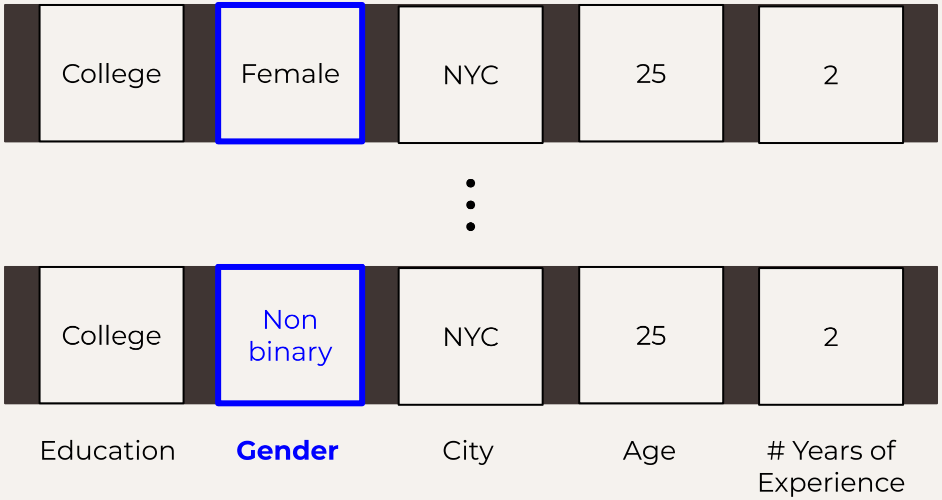


How many clones?

The number of distinct values in the ***sensitive feature's*** input bounds

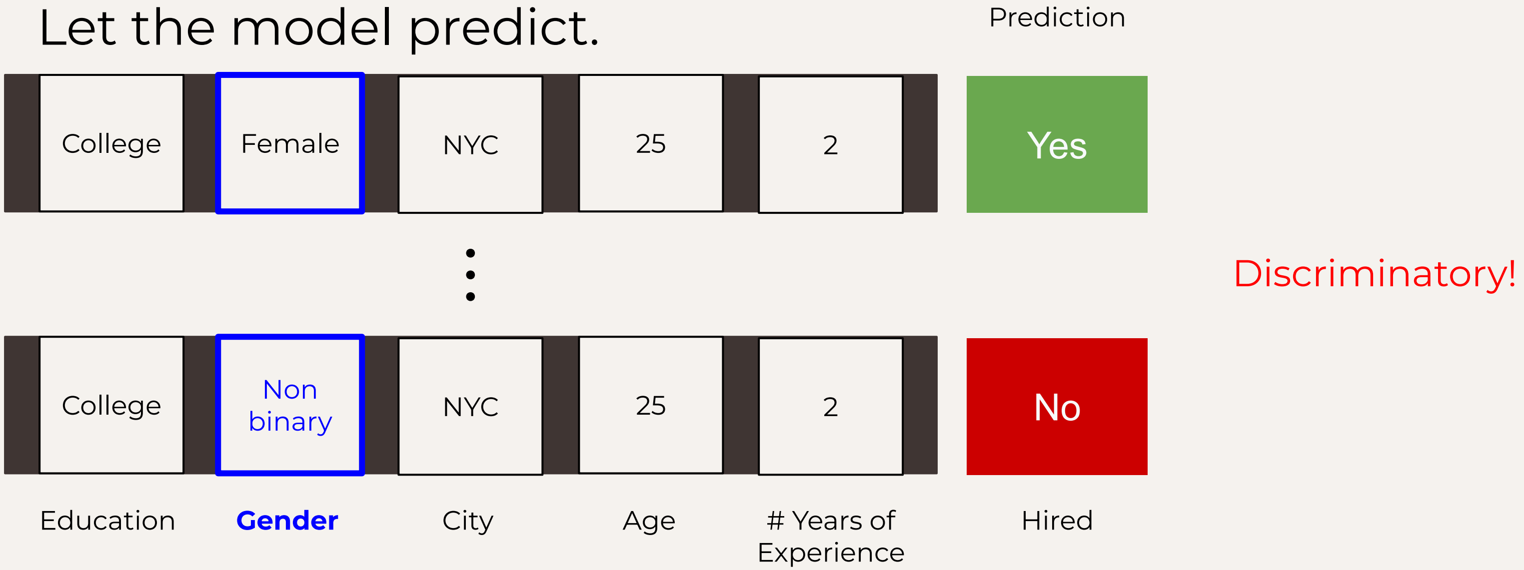
How do we evaluate fairness?

1. Clone the input.
2. Only change the value of the **sensitive feature** to the different possible values.

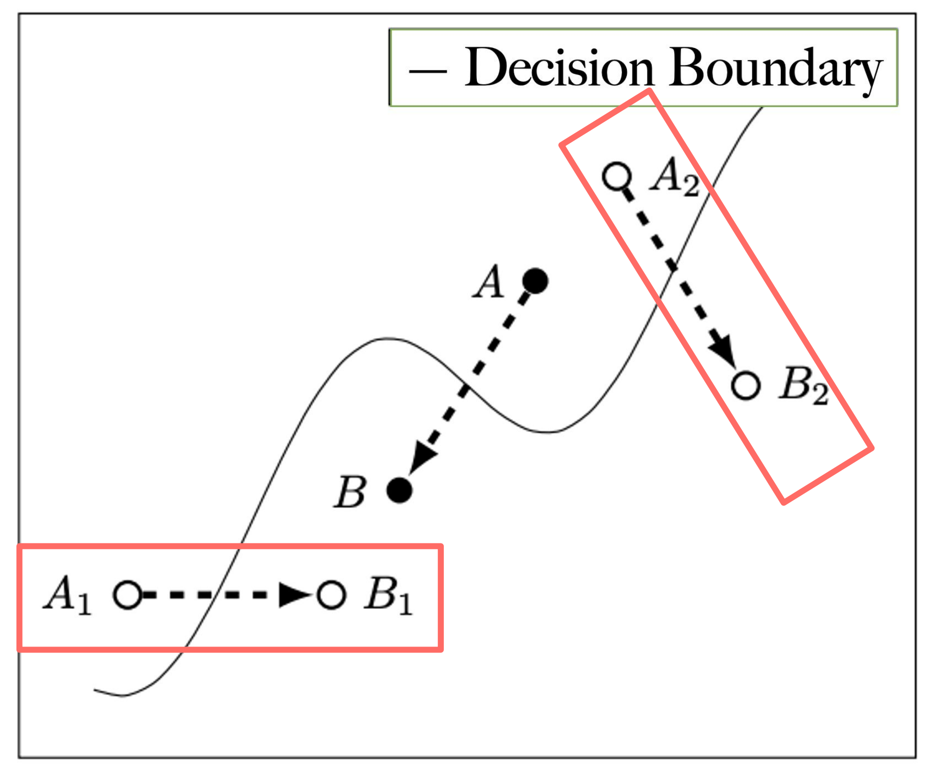


How do we evaluate fairness?

1. Clone the input.
2. Only change the value of the sensitive parameter to the different possible values.
3. Let the model predict.

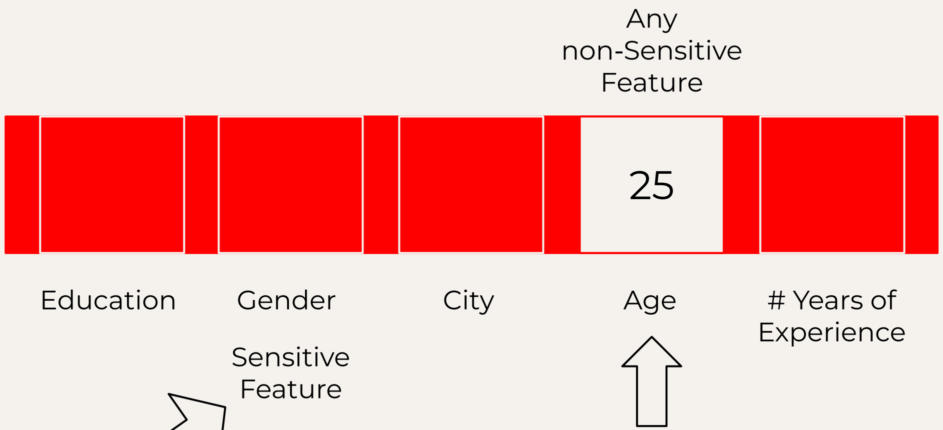


Local Search



What does ‘perturbing’ an input mean?

Modifying the
globally collected discriminatory input
slightly to find another
potentially discriminatory input

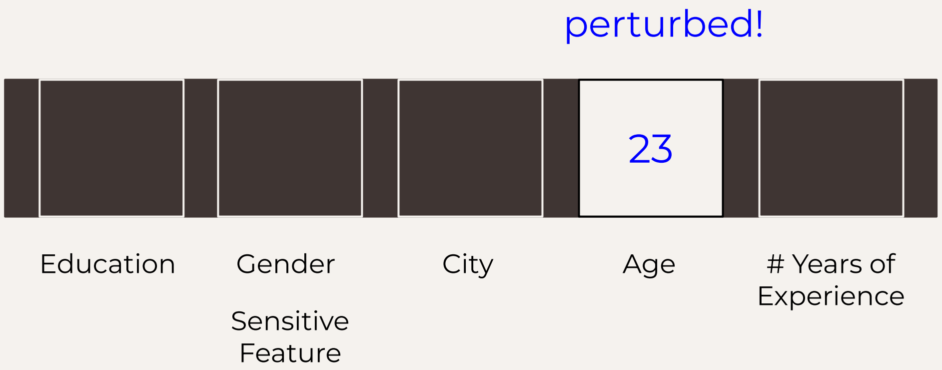


Keep this the same!

We expect the outcome to be similar to the original , thanks to **robustness**

What does ‘perturbing’ an input mean?

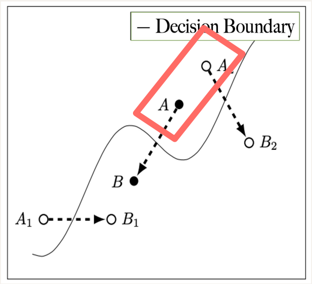
Modifying the globally collected discriminatory input *slightly* to find another *potentially* discriminatory input



And now we do the clone and compare method again on this input

The reason we’re doing this is because we want to search efficiently

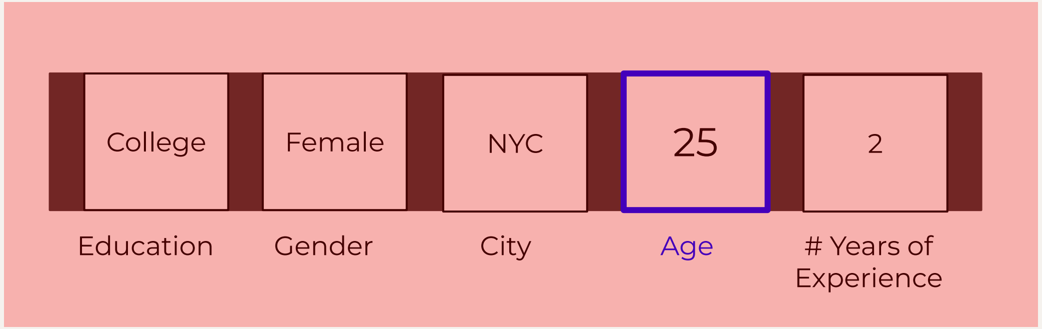
College	Female	NYC	25	2
Education	Gender	City	Age	# Years of Experience



Which non-sensitive feature, when perturbed, will most likely result in **another discriminatory input?**

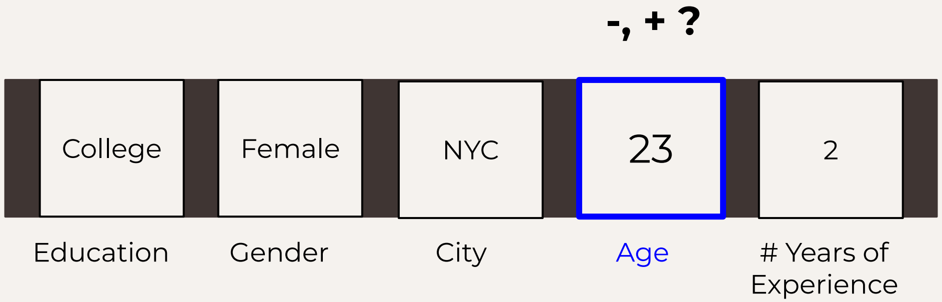
Does *increasing* or *decreasing* the that feature result in another discriminatory input?

The reason we're doing this is because we want to search efficiently



If the perturbed input is discriminatory.. **increase** the probability of choosing this feature for our next perturbation

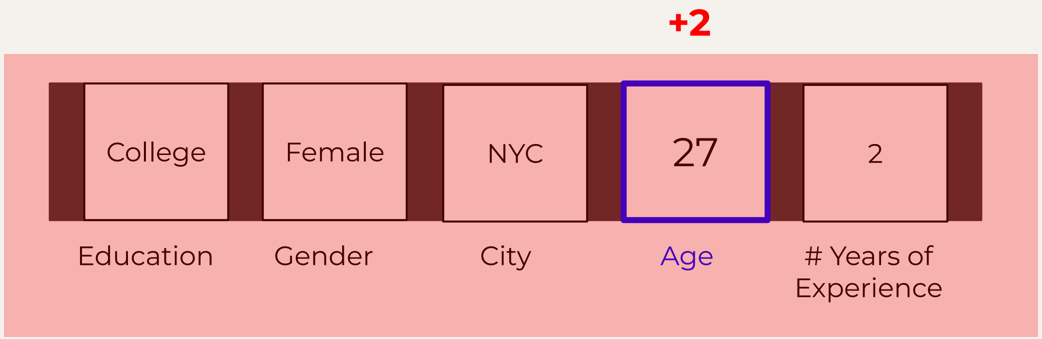
The reason we're doing this is because we want to search efficiently



Does *increasing* or *decreasing* the age result in another discriminatory input?

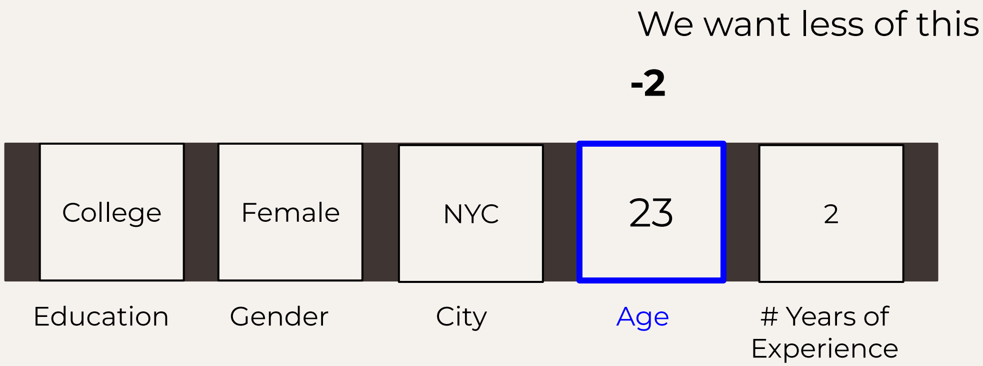
The reason we're doing this is because we want to search efficiently

We want more of this!



This perturbed input is discriminatory
⇒ **increase** the probability of choosing this direction

The reason we're doing this is because we want to search efficiently



This perturbed input is <u>not</u> discriminatory input
⇒ **decrease** the probability of choosing this direction

In Summary

Local Search is where Aequitas tries to be **smart** in collecting discriminatory inputs by directing its perturbation in a way that will discover the **most new discriminatory inputs**

Retraining

1. Add a **small portion** of the retraining data to the original dataset and train the model
2. If biasedness **decreases**, go back to step 1
3. If biasedness **increases**, terminate

3
Our Work

Initial State of Aequitas

The authors of the original paper released Aequitas as a proof of concept implementation.

It could run Aequitas on a dataset provided by the user.

Improvements - Initial State of Aequitas

Major limitations we needed to address

- Needed modularization
- Only allowed for **a singular**, **binary** sensitive feature
- Hard-coded for the above case
- **Slow**

Modularization

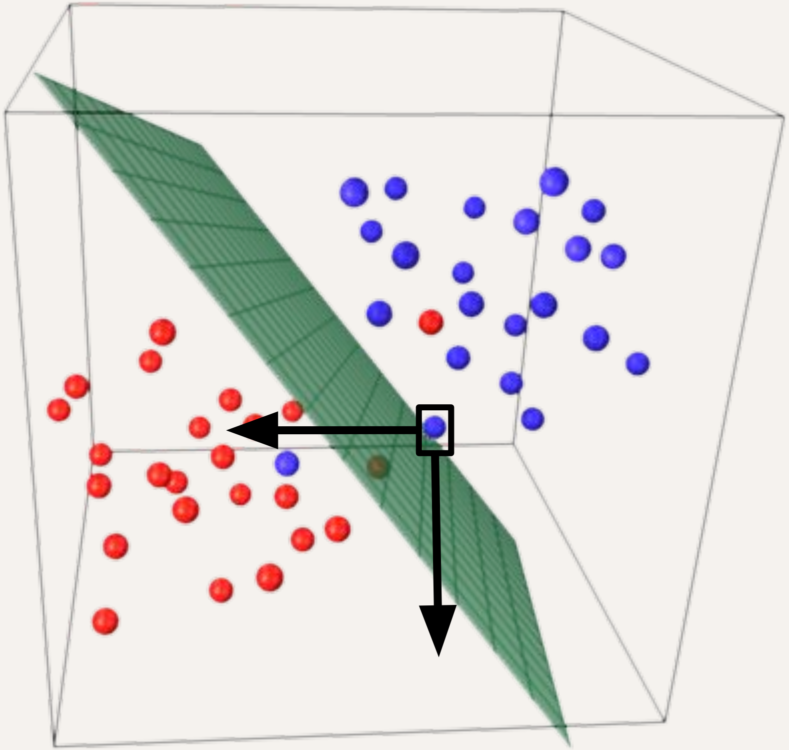
- **Solution:** We packaged Aequitas into a Python package that can now be installed by running:
    - pip install Phemus

Hi

Modularization

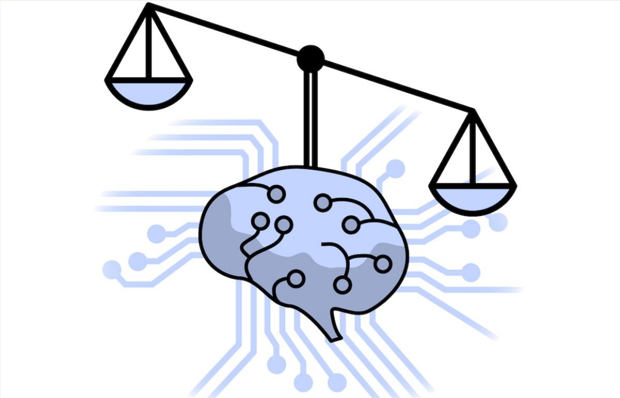
```
from Phemus import *

generate_sklearn_classifier('''parameters goes here''')
aequitas_random_sklearn('''parameters goes here''')
retrain_sklearn('''parameters goes here''')
```

Only allows binary sensitive feature



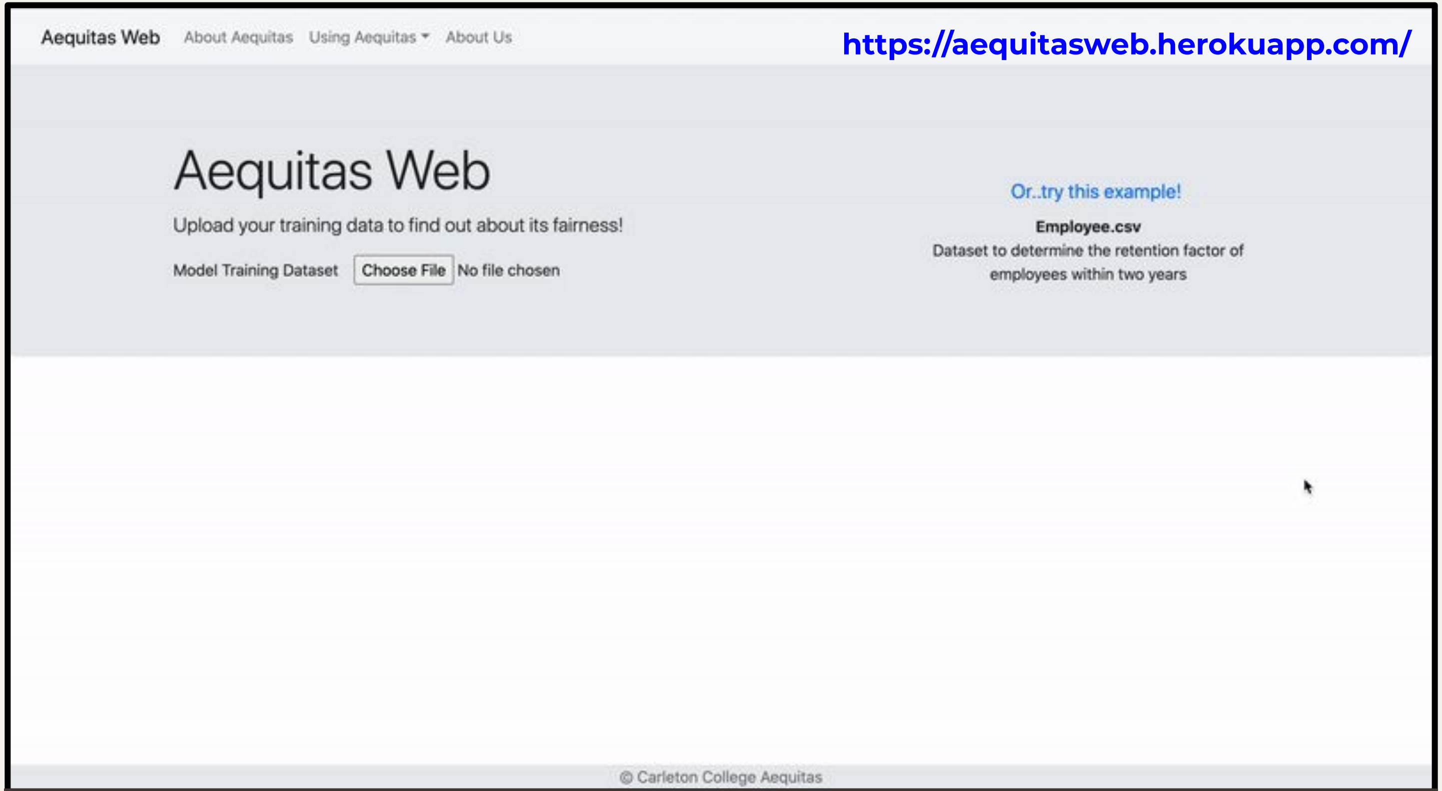
Only allows one sensitive feature

- **Solution:** Run Aequitas multiple times on the same dataset
- Problem: Does not take into account how multiple features can interact with each other.



Multiprocessing

- The original implementation executed all of the implementations in **one thread**.
- Unsustainable for **multi-dimensional** sensitive features.
- Leverage **multiprocessing module** in Python 3.
- We decided to **split the work** in local search across multiple processes.



Django backend | React frontend | Heroku hosting

4
Conclusion

Significance

What can be done from here?

What does this all mean for machine learning fairness?

References.

Udeshi, S., Arora, P., & Chattopadhyay, S. (2018). Automated directed fairness testing. *ASE 2018 - Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, 98–108. https://doi.org/10.1145/3238147.3238165

Zafar, M. B., Valera, I., Rodriguez, M. G., & Gummadi, K. P. (2015). *Fairness Constraints: Mechanisms for Fair Classification*.

Russell, S., & Norvig, P. (2020). *Artificial Intelligence: a Modern Approach* (4th ed.). Pearson.

Lam, L., & Suen, C. Y. (1997). Application of majority voting to pattern recognition: An analysis of its behavior and performance. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.* https://doi.org/10.1109/3468.618255

Hutchinson, B., & Mitchell, M. (2019). 50 Years of Test (Un)fairness: Lessons for machine learning. *FAT* 2019 - Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, 49–58. https://doi.org/10.1145/3287560.3287600

Galhotra, S., Brun, Y., & Meliou, A. (2017, August 21). Fairness testing: testing software for discrimination. *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. https://doi.org/10.1145/3106237.3106277

References

Fawzi, A., Fawzi, O., & Frossard, P. (2015). *Analysis of classifiers' robustness to adversarial perturbations*. http://arxiv.org/abs/1502.02590

Dua, D., & Graff, C. (2017). *UCI Machine Learning Repository: Haberman's Survival Data Set*. http://archive.ics.uci.edu/ml

Dua, D., & Graff, C. (2017). *UCI Machine Learning Repository: Banknote Authentication Data Set*. http://archive.ics.uci.edu/ml

Dobrow, R. P. (2013). *Probability with applications in R*.

Thanks

Topics We Covered

1
Introduction
Machine Learning &
Machine Learning Fairness

2
Literature Review
How Aequitas Works
Implementation

3
Our Work
Usability Improvements

4
Conclusion
Where to go from here

Questions?