Black-Box Approaches to Fair Machine Learning

Samuel Yeom

CMU-CS-21-121

June 2021

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Matt Fredrikson, Chair
Alexandra Chouldechova
Anupam Datta
Ariel Procaccia
Sharad Goel, Stanford University

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.

This research was sponsored by National Science Foundation award numbers CNS1704845 and CNS1943016, and by United States Air Force Research Laboratory award number FA87501520277. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: proxy use, optimal transport, FlipTest, individual fairness, randomized smoothing, distribution shift, underrepresentation, membership inference, shadow model

Abstract

As machine learning models are increasingly used to make consequential decisions involving humans, it is important that the models do not discriminate on the basis of protected attributes such as race and gender. However, the model holder are not the people who bear the brunt of the harm from a discriminatory model, so there is little natural incentive for the model holder to fix a discriminatory model. It would thus be societally beneficial if other entities could also detect or mitigate unfair behavior in these models. *Black-box* methods, which require only query access to the model, are well suited for this purpose, as they are feasible to carry out without knowing the full details of the model.

In this thesis, I consider three different forms of unfairness and present black-box methods that address them. The first of the three is *proxy use*, where some component of a model is a proxy for a protected attribute. The second is the lack of *individual fairness*, which formalizes the intuitive notion that a model should not make arbitrary decisions. Finally, a model may have an *unrepresentative training set*, which can lead the model to exhibit varying degrees of accuracy for different protected groups. For each of these behaviors, I propose one or more methods that can help detect such behavior in models or ensure the lack thereof. These methods require only black-box access to the model, allowing them to be effective even when the model holder is uncooperative. My theoretical and experimental analysis of these methods evidence their effectiveness in this setting, showing that they are useful technical tools that can support an effective response against discrimination.

Acknowledgments

First of all, I would like to thank my advisor Matt Fredrikson for the copious research ideas and career advice that he has given me throughout all five of my years at CMU. None of the work presented in this thesis would have been possible without him. I am also grateful to my other co-authors (in alphabetical order) Emily Black, Anupam Datta, Irene Giacomelli, Marc Juarez, Somesh Jha, Alan Menaged, Ameet Talwalkar, Zilong Tan, and Michael Carl Tschantz, who have been indispensable to my research as well. Finally, I would like to thank the thesis committee—Matt Fredrikson, Alexandra Chouldechova, Anupam Datta, Ariel Procaccia, and Sharad Goel—for agreeing to serve on the committee and for their helpful feedback on this thesis.

Outside of work, I've had the pleasure of co-leading the Puzzle Hunt CMU student club. Writing a puzzle hunt every semester takes a lot of effort, and I would like to thank all past and present members of the club for making this possible. I am also thankful to my former officemate Abutalib Aghayev for his ability to troubleshoot technological issues and for our daily ritual of solving the New York Times crosswords together. Last but not least, I would like to thank my parents for their unwavering love and support.

Contents

Chapter 1

Introduction

The use of machine learning in insurance [72], criminal justice [36], and child welfare [99] raises concerns about fairness, as decisions based on model predictions may discriminate on the basis of legally protected demographic attributes, such as race and gender. These concerns are driven by high-profile examples of models that appear to have discriminatory effect, ranging from gender bias in job advertisements [28] and hiring [27] to racial bias in same-day delivery services [49] and predictive policing [6].

Meanwhile, laws and regulations in various jurisdictions prohibit certain practices that have discriminatory effect, regardless of whether the discrimination is intentional. For example, the U.S. has recognized the doctrine of *disparate impact* since 1971, when the Supreme Court held in *Griggs v. Duke Power Co.* [89] that the Duke Power Company had discriminated against its Black employees by requiring a high-school diploma for promotion when the diploma had little to do with competence in the new job. In response, the Equal Employment Opportunity Commission promulgated the four-fifths rule [35], which is a guideline that states that the rate at which people receive a favorable outcome (e.g., being hired or promoted) should be roughly equal across legally protected demographic groups. The intuition behind this guideline has since been formalized in the fair machine learning literature as *statistical parity*.

One benefit of this definition—and perhaps one of the reasons why the EEOC chose the four-fifths rule—is that anyone, even an outsider who does not have detailed knowledge of the decision-making process, can easily apply it in practice. Because statistical parity only checks for demographic balance in the outcome, simply querying a model on the various demographic groups is sufficient to verify statistical parity. Therefore, a statistical parity audit requires only *black-box* access to the model. In practice, companies often sell query access to their models, so we do not need any additional cooperation from the companies to carry out this audit.

This is useful because the model holders are usually not the people who bear the brunt of the harm from a discriminatory model. For example, when a bank wrongfully denies a loan to an applicant due to a faulty credit scoring model, the bank loses only a minuscule portion of its revenue, but the loan applicant loses the ability to make a major purchase. As a result, the loan applicant would be more eager to fix the discriminatory model than the bank. A black-box audit enables these loan applicants or their advocates to detect the unfair behavior in the model, pressuring the model holder into fixing the discrimination that would not have been fixed otherwise.

Furthermore, black-box techniques have an additional benefit that it is model-agnostic and thus generally applicable to a wide variety of models. Some complex models, such as neural networks, are difficult for humans to interpret even with unrestricted white-box access. Model-agnostic black-box techniques can bring even these models into compliance with relative ease.

Insufficiency of Parity. Although widely applied, statistical parity is not sufficient for a model to be fair,[1] so it is desirable to have black-box methods that can verify or enforce other aspects of fairness as well. Below, I list some ways in which a model can be blatantly unfair while technically satisfying statistical parity. The examples below are exaggerated to clearly portray the unfair behaviors, but less extreme forms of these behaviors can arise naturally without any malicious intent.

1. **Incomplete Repair.** Suppose that an employer uses a racially biased model to determine which of their employees will be promoted. After noticing the racial bias, the employer attempts to "repair" the outcome by arbitrarily promoting some more members of the disadvantaged race until statistical parity is reached. The resulting procedure is still unfair to individual members of the disadvantaged race who were not eventually promoted. In fact, this is why the U.S. Supreme Court ruled that statistical parity is not a complete defense to claims of disparate impact [90].
2. **Arbitrary Predictions.** Consider a binary classifier that flips a coin to determine its output, paying no attention to its input except for ensuring that the same input always receives the same output.[2] This model satisfies statistical parity because, for each group, exactly half of its members are expected to receive the favorable outcome. However, although there are some situations in which the distribution of scarce resources by lottery is appropriate, this behavior is generally considered arbitrary and unfair if there is an expectation that more qualified people should receive more favorable outcomes.
3. **Accuracy Disparity.** Suppose that a bank uses a model to determine which of their loan applicants are creditworthy. Although receiving a loan is in general a favorable outcome, defaulting on the loan is detrimental. Therefore, even if the model approves loans at the same rate for two protected groups, if it is not as accurate in predicting whether members of one group will eventually repay the loan, ultimately the model will disproportionately harm members of that group.

In this thesis, I describe several black-box methods for detecting or addressing these unfair behaviors. Due to the black-box nature of these methods, they are available to anyone with query access who is concerned about how another entity has deployed a model. The rest of this chapter summarizes these methods and explains how they relate to the above examples of unfairness. Overall, my contributions in this thesis are novel and represent significant advances in the fundamental scientific understanding of fair machine learning.

[1] As a side note, statistical parity is not *necessary* for a fair model either. For example, a hiring model for a job that requires heavy physical labor is expected to hire more men than women. Moreover, simply enforcing statistical parity without regard for possible justifications for a disparity may be prohibited on the grounds of intentional discrimination [91].

[2] In other words, this model behaves like a random oracle [12] in cryptography.

Before I proceed with the summary, I would like to remind the reader that a comprehensive response against discrimination requires more than just technical solutions. Although the methods in this thesis can flag some aspects of a model for further review, they cannot by themselves determine whether a model is fair (e.g., see Footnote 1). For this reason, it is important to consult domain experts and the affected community when machine learning models are applied to human subjects.

1.1 Incomplete Repair

In Chapter 2, I address the problem of proxy use [30], which is a formalization of the intuition that some component of the model may be a proxy for a protected attribute even if the model taken as a whole appears to be fair. For example, in the first example above, the original biased model and the subsequent incomplete repair can both be considered proxies for race. However, prior work on proxy use [31] has axiomatically proven that black-box access is insufficient for proxy detection. Therefore, I first address the linear regression setting, where there is no meaningful difference between black-box and white-box audits because Gaussian elimination can easily reveal the regression coefficients. I use these coefficients to construct a relatively simple optimization procedure that is guaranteed to find a proxy if one exists [106].

More complex models are outside the scope of this optimization procedure, but we can create a black-box procedure that approximates the detection of proxies. The key idea behind this procedure, called FlipTest [13], is that a model should give the same output to two people if they differ only in the protected attribute. However, simply flipping the protected attribute is not sufficient because models often do not use the protected attribute directly, instead learning to discriminate through the use of proxies. To address this issue, I apply an optimal transport mapping [100] to find an individual's counterpart, who is similar to the individual but differs in the protected attribute and possibly other input features as well. The optimal transport mapping determines how much to shift the other input features based on the different distributions of the protected groups.

FlipTest then checks whether the model gives the same output for the individual and the counterpart. By aggregating these individual results, we can learn which subgroups may be experiencing discrimination and which features may be causing the discrimination. The experimental results show that the output of FlipTest is often in alignment with, but distinct from, that of black-box causal analysis based on the quantitative input influence (QII) [29]. Thus, FlipTest can supplement existing black-box auditing techniques, in addition to serving as a more efficient approximation of causal analysis.

1.2 Arbitrary Predictions

Black-box proxy detection methods mitigate many of the shortcomings of statistical parity when the main concern is discrimination with respect to a single, well-defined protected attribute. However, it is also important that the model behavior is not arbitrary, both within a protected group and across different protected groups. Chapter 3 thus explores individual fairness [34],

which captures the notion that similar people should be treated similarly by imposing a continuity requirement on models. In particular, I focus on black-box analysis, which has the benefit that it is model-agnostic.

One criticism of individual fairness is that it is difficult to apply in practice because it requires one to define which individuals are "similar" to each other [23]. To get around this issue, rather than ascertaining the fairness of a model given a predetermined similarity metric, I find a metric for a given model that satisfies individual fairness [105]. This metric can then guide the discussion on whether the model is fair. Like in the previous chapter, I first analyze the case of linear regression and then find a way to handle the general case. For linear models, I prove that the metric that best characterizes the model behavior is derived from the regression coefficients.

By contrast, more complex models are harder to analyze, and some, such as neural networks, tend not to be individually fair under any reasonably sized metric. Nevertheless, these models can be more accurate than linear models, so it would be desirable to modify them so that they are fair. I show that this is indeed possible, proving that randomized smoothing [25] can make any classifier individually fair under any given weighted L^p metric. Randomized smoothing requires only black-box access to the classifier, so any complex model to which we have query access can be smoothed to have the same individual fairness guarantees as a given linear regression model. Moreover, my experiments show that there is a trade-off between fairness and utility for randomized smoothing, and we can control how much utility to sacrifice for fairness.

1.3 Accuracy Disparity

Finally, in Chapter 4 I address the issue that a model can exhibit varying degrees of accuracy on different protected groups. An intuitive reason for this accuracy disparity is that a group is underrepresented in the training set, leaving the model with insufficient examples to learn patterns specific to that group. However, underrepresentation is not the only possible reason—the choice of a learning algorithm [46] or features [20] can also lead to an accuracy disparity. At the same time, it is important to identify the cause of an accuracy disparity, as the remedy for the disparity depends on its cause. However, if the model holder is uncooperative, it is difficult to detect an unrepresentative training set because the training set is often proprietary.

My contribution here is a black-box audit for detecting an unrepresentative training set, which I call a *group distribution shift*. My formal definition of the group distribution shift audit [52] is based on the membership inference attack [107], which tries to gain information about whether a given input is a part of the model's training set. By modifying the shadow model technique [87] previously used for membership inference, I arrive at an auditing procedure that specifically detects group distribution shifts. My experimental results show that this audit can indeed detect unrepresentative training sets, performing significantly better than a naive audit that treats any accuracy disparity as evidence of underrepresentation.

In addition, I address a more general phenomenon of *distribution shift*, which occurs when a model's training distribution differs from the distribution encountered during model deployment. My experimental results also show that a slight change to the group distribution shift audit gives it the ability to distribution shifts in general. This audit may be useful in malware or intrusion detection settings, where distribution shift can enable attackers to bypass security measures.

1.4 Summary

This thesis is a compilation of black-box techniques for detecting or mitigating various forms of unfairness in machine learning models. Chapter 2 begins with a method that detects proxies for a protected attribute in a linear regression model [106]. Then, for more general models, I present a black-box technique called FlipTest [13] that approximates the existence of proxies. In Chapter 3, I show how to apply the black-box technique of randomized smoothing [25] to ensure the individual fairness of a model under a given metric [105]. Finally, Chapter 4 formalizes an audit for distribution shift, which includes the case where a model's training set is unrepresentative. Following the black-box shadow model approach [87], I design an auditing procedure that ascertains whether the training set of a given model conforms to a normative distribution [52].

Chapter 2

Detecting Proxy Use

The incomplete repair example in Chapter 1 demonstrates that a model that appears to be fair can still hide an obviously unfair subprocedure. Prior work on *proxy use* [30] addresses this issue by considering how the model arrives at its output. A proxy for a protected attribute is defined as a portion of the model that is both causally influential on the model's output and statistically associated with the protected variable. For an example of a proxy, recall the incomplete repair example from Chapter 1, where there is an attempt to make a biased model fair by arbitrarily changing the model outcome for some members of the disadvantaged group. Here, the original biased model is a proxy for a protected demographic attribute, indicating the presence of discriminatory behavior in the "repaired" model.

In prior treatment of proxy use [31], it was proven that black-box access is not always sufficient to ascertain the presence of proxies. On the other hand, a black-box proxy detection procedure would allow an audit from an external entity, who may be more motivated than the model holder to uncover potentially discriminatory model behavior. Therefore, in Section 2.2 I first consider the setting of linear regression, where there is no meaningful difference between black-box and white-box access because Gaussian elimination can easily reveal the regression coefficients.

Section 2.2.1 reviews the original definition of proxy use by Datta et al. [30] for classification models and then shows how to modify this definition to get one that is applicable to the setting of linear regression. This new definition is justified axiomatically in Section 2.2.2. While the previous notion of proxy use is prohibitively expensive to apply at scale to real-world models [30], the new definition admits a convex optimization procedure that leads to an efficient detection algorithm, presented in Section 2.2.3. In Section 2.2.4, I evaluate this algorithm with two real-world predictive policing applications, finding that the algorithm, despite taking little time to run, accurately identifies parts of the model that are the most problematic in terms of discrimination.

Section 2.3 describes FlipTest [13], which is a black-box approximation of the proxy detection procedure for any binary classifier. FlipTest poses a question that focuses on the data rather than the model: Had an individual been of a different protected status, would the model have treated the individual differently? This approach resembles that of Kusner et al. on *counterfactual fairness* [61], which uses a generative causal model to answer the same question. However, causality is in general difficult to ascertain, and the use of an inaccurate causal model may lead to incorrect conclusions. Furthermore, the legal frameworks governing discrimination (e.g., dis-

parate impact [89] in the US) do not require a causal model, so tests based on counterfactual fairness may fail to identify instances of legally actionable discrimination.

By contrast, FlipTest leverages optimal transport [100] to match individuals in different protected groups, creating similar pairs of in-distribution samples. Then, if the model gives different outputs for the two individuals in a pair, the pair is flagged for further review. Although one pair by itself is not a strong evidence of discrimination, by aggregating these pairs we can learn which subgroups may be experiencing discrimination and which features may be responsible for the discrimination. The results of the experiments in Section 2.3.4 show that FlipTest is indeed capable of detecting some proxies.

However, FlipTest does not use causal information at all, so it alone is not sufficient to ascertain the presence of proxies, which are defined partly in terms of causal influence.[3] Therefore, in Section 2.4 I supplement FlipTest with causal analysis to create a pipeline that performs a more holistic audit. Although the original proxy detection algorithm by Datta et al. [30] can be used for this purpose, it requires white-box access to the model. Thus, to preserve the black-box nature of the auditing pipeline, I instead compute the quantitative input influence (QII) measure [29] to quantify the causal influence. The experimental results are mostly consistent with those of the transparency reports, but sometimes they provide new insights that aid in understanding the behavior of the models.

The contents of this chapter are largely based on two previously published works [13, 106].

2.1 Related Work

The concept of proxy use under consideration in this chapter is based on the work of Datta et al. [30]. Their proxy detection algorithm identifies a biased portion of a model that is causally influential in the model's final output. By contrast, Kilbertus et al. [58] also use causal information, but through a causal model [75] that is separate from the model in question. An alternative measure of proxy strength has been proposed by Adler et al. [2], who define a single real-valued metric called indirect influence. As I show in Section 2.2.3, the two-metric-based approach of Datta et al. leads to an efficient proxy detection algorithm for linear regression models.

Many other methods in fair machine learning also consult counterfactual information. Kusner et al. [61] compare the model's behavior on a real input and a counterfactual, causally generated input. Similarly, Wachter et al. [101] generate simple L^1-nearest counterfactuals as a form of explanations for model outputs, and Ustun et al. [98] develop a method that outlines what actions individuals can take to change their classification outcome in linear models. However, these methods generate potentially unrealistic, out-of-distribution points, which can jeopardize their conclusions. By contrast, FlipTest has the advantage that the points it generates conform to the data distribution.

One application of FlipTest is uncovering *subgroup unfairness* [44, 57], i.e., identifying subgroups that are possibly harmed as a result of their group membership. FairTest [97] uses a decision tree to find subgroups with high discriminatory association, while taking care to ensure

[3]This notion of causality is distinct from the causal models used in counterfactual fairness. Causal influence in proxy use relates to how the model computes its output from its inputs. By contrast, causal models describe the causal relationships between the inputs.

that the association is statistically significant. However, as I show in Section 2.3.4, FairTest does not handle the case where the input feature itself is biased. Zhang et al. [115] also find a computationally faster way to search the exponentially large number of subgroups, but their method also suffers from the same issue that FairTest does. Kearns et al. [56] prove that checking for subgroup fairness is equivalent to weak agnostic learning, which is computationally hard in the worst case. FlipTest differs from these works in that it does not require the subgroups of interest to be specified before the fairness testing.

Like FlipTest, some other works also propose using the optimal transport mapping in the context of fairness. Del Barrio et al. [32] use the optimal transport mapping to extend a previous method [37] of data pre-processing, which obfuscates protected attribute information, to the multivariate setting. Yang et al. [104] also develop a method of approximating an (unbalanced) optimal transport mapping using GANs. Their formulation is closely related to FlipTest's, but they do not consider its application to fairness testing. Altschuler et al. [3] and Quanrud [78] present efficient methods for approximating the optimal transport mapping, but the resulting mappings are not defined for previously unseen data points. By contrast, the mappings produced by Leygonie et al. [62], Perrot et al. [77], and Seguy et al. [84] do generalize to unseen points, making them suitable for use with FlipTest.

2.2 Proxy Use in Linear Regression Models

I now begin by examining the feasibility of detecting proxies in linear models. The implementation of a linear regression model can be reconstructed efficiently with black-box access by using Gaussian elimination. We will thus assume that the auditor has access to the regression coefficients.

2.2.1 Definitions

We work in the standard machine learning setting, where a model is given the values of several input features that correspond to a data point. I write $\mathcal{X} = (X_1, \ldots, X_d)$ to denote the input, where $X_1, \ldots, X_d$ are random variables. Then, the output of a linear regression model $\hat{Y}$ can be written as $\beta_1 X_1 + \cdots + \beta_d X_d$, where β_i represents the coefficient for the input variable X_i.

In the case where each data point represents a person, care must be taken to avoid discrimination on the basis of a protected demographic attribute, such as race or gender. Let us denote such protected attribute by the random variable Z. In practice, Z is usually binary (i.e., $Z \in \{0, 1\}$), but the results in Section 2.2 apply to arbitrary numerical random variable Z.

Proxy Use in Prior Work

Previously, Datta et al. [30] defined proxy use of a random variable Z as the presence of an intermediate computation in a program that is both statistically *associated* with Z and causally *influential* on the final output of the program. Instantiating this definition to a particular setting therefore entails specifying an appropriate notion of "intermediate computation", a statistical association measure, and a causal influence measure.

Datta et al. identify intermediate computations in terms of syntactic decompositions into *subprograms* P, $\hat{Y}'$ such that $\hat{Y}(\mathcal{X}) \equiv \hat{Y}'(\mathcal{X}, P(\mathcal{X}))$. Then, the association between P and Z is given by an appropriate measure such as mutual information, and the influence of P on $\hat{Y}$ is defined as shown in Eq. (2.1), where $\mathcal{X}$ and $\mathcal{X}'$ are drawn independently from the population distribution.

$$\text{Infl}_{\hat{Y}}(P) = \Pr_{\mathcal{X},\mathcal{X}'}[\hat{Y}(\mathcal{X}) \neq \hat{Y}'(\mathcal{X}, P(\mathcal{X}'))], \tag{2.1}$$

Intuitively, influence describes likelihood that an independent change in the value of P will cause a change in $\hat{Y}$. This makes sense for classification models because a change in the model's output corresponds to a change in the predicted class of a point, as reflected by the frequent use of 0-1 loss in the classification setting. On the other hand, regression models have real-valued outputs, so the square loss is more appropriate for these models. Therefore, I transform Eq. (2.1), which is simply the expected 0-1 loss between $\hat{Y}(\mathcal{X})$ and $\hat{Y}'(\mathcal{X}, P(\mathcal{X}'))$, into Eq. (2.2), which is the expected square loss between these two quantities.

$$\mathbb{E}_{\mathcal{X},\mathcal{X}'}[(\hat{Y}(\mathcal{X}) - \hat{Y}'(\mathcal{X}, P(\mathcal{X}')))^2] \tag{2.2}$$

Before we can reason about the suitability of this measure, we must first define an appropriate notion of intermediate computation for linear models.

Linear Components

The notion of subprogram used for discrete models [30] is not well suited to linear regression. To see why, consider the model $\hat{Y} = \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3$. Suppose that this is computed using the grouping $(\beta_1 X_1 + \beta_2 X_2) + \beta_3 X_3$ and that the definition of subprogram honors this ordering. Then, $\beta_1 X_1 + \beta_2 X_2$ would be a subprogram, but $\beta_1 X_1 + \beta_3 X_3$ would not be even though $\hat{Y}$ could have been computed equivalently as $(\beta_1 X_1 + \beta_3 X_3) + \beta_2 X_2$. Therefore, it is desirable to allow any subset of the terms used in the model to define a subprogram, capturing the commutativity and associativity of addition. However, this definition still excludes expressions such as $\beta_1 X_1 + 0.5\beta_3 X_3$, which may be a stronger proxy than either $\beta_1 X_1$ or $\beta_1 X_1 + \beta_3 X_3$. Thus, I use Definition 2.1 to characterize intermediate computations in the setting of linear regression.

Definition 2.1 (Component)**.** *Let $\hat{Y} = \beta_1 X_1 + \cdots + \beta_d X_d$ be a linear regression model. A random variable P is a* component *of $\hat{Y}$ if and only if there exist $\alpha_1, \ldots, \alpha_d \in [0, 1]$ such that $P = \alpha_1 \beta_1 X_1 + \cdots + \alpha_d \beta_d X_d$.*

Linear Association and Influence

Having defined a component as the equivalent of a subprogram in a linear regression model, I now formalize the association and influence conditions given by Datta et al. [30].

A linear model only uses linear relationships between variables, so our association measure only captures linear relationships. In particular, we use the Pearson correlation coefficient, and we square it so that a higher association measure always represents a stronger proxy. Note that $\text{Asc}(P, Z) \in [0, 1]$, with 0 representing no linear correlation and 1 representing a fully linear relationship.

Definition 2.2 (Association)**.** *The* association *of two nonconstant random variables P and Z is defined as* $\mathrm{Asc}(P, Z) = \frac{\mathrm{Cov}(P,Z)^2}{\mathrm{Var}(P)\mathrm{Var}(Z)}$.

To formalize influence, we continue from Eq. (2.2). Definition 2.1 gives us $\hat{Y}(\mathcal{X}) = \sum_{i=1}^{d} \beta_i X_i$ and $\hat{Y}'(\mathcal{X}, P(\mathcal{X}')) = \sum_{i=1}^{d} (1-\alpha_i)\beta_i X_i + \alpha_i \beta_i X_i'$. Substituting these into Eq. (2.2) gives

$$\begin{aligned}
\mathbb{E}_{\mathcal{X},\mathcal{X}'}[(\hat{Y}(\mathcal{X}) - \hat{Y}'(\mathcal{X}, P(\mathcal{X}')))^2] &= \mathbb{E}_{\mathcal{X},\mathcal{X}'}[(\textstyle\sum_{i=1}^{d} \alpha_i\beta_i X_i - \alpha_i\beta_i X_i')^2] \\
&= \mathbb{E}_{\mathcal{X},\mathcal{X}'}[(P(\mathcal{X}) - P(\mathcal{X}'))^2] \\
&= \mathrm{Var}(P(\mathcal{X}) - P(\mathcal{X}')) \\
&= \mathrm{Var}(P(\mathcal{X})) + \mathrm{Var}(P(\mathcal{X}')),
\end{aligned}$$

which is proportional to $\mathrm{Var}(P(\mathcal{X}))$ since $\mathcal{X}$ and $\mathcal{X}'$ are i.i.d. Definition 2.3 captures this reasoning, normalizing the variance so that $\mathrm{Infl}_{\hat{Y}}(P) = 1$ when $P = \hat{Y}$ (i.e., $\alpha_1 = \cdots = \alpha_d = 1$).

Definition 2.3 (Influence)**.** *Let P be a component of a linear regression model $\hat{Y}$. The* influence *of P is defined as* $\mathrm{Infl}_{\hat{Y}}(P) = \frac{\mathrm{Var}(P)}{\mathrm{Var}(\hat{Y})}$.

When it is obvious from the context, the subscript $\hat{Y}$ may be omitted. Note that influence can exceed 1 because the input features can cancel each other out, leaving the final model less variable than some of its components.

Finally, Definition 2.4 formally characterizes proxy use in linear regression models in terms of the above association and influence measures.

Definition 2.4 ((ϵ, δ)-Proxy Use)**.** *Let $\epsilon, \delta \in (0, 1]$. A model $\hat{Y} = \beta_1 X_1 + \cdots + \beta_d X_d$ has* (ϵ, δ)-proxy use *of Z if there exists a component P such that* $\mathrm{Asc}(P, Z) \geq \epsilon$ *and* $\mathrm{Infl}_{\hat{Y}}(P) \geq \delta$.

2.2.2 Axiomatic Justification of the Influence Measure

I now present an axiomatic justification for Definition 2.3, arguing that variance is the natural metric for quantifying the influence of a component. More specifically, I prove that variance is the unique function, up to a multiplicative constant, that satisfies a few desirable properties of an influence measure.

While the influence measure was motivated by the need to characterize the behavior of some component P of model $\hat{Y}$, it can quantify the behavior of general random variables as well. Therefore, in this section I work in the general setting where P need not be a component of $\hat{Y}$. In particular, this means that the subscript $\hat{Y}$ in the notation $\mathrm{Infl}_{\hat{Y}}(P)$ will be omitted. Note that, if we restrict P to be a component of $\hat{Y}$, sometimes (e.g., when $\hat{Y}$ takes in only one input) it is impossible to impose meaningful restrictions on the influence measure with natural axioms such as the ones I present here:

Nonnegativity. $\mathrm{Infl}(P) \geq 0$ for all P.

Nonconstant positivity. $\mathrm{Infl}(P) = 0$ if and only if P is a constant.

Additivity. $\mathrm{Infl}(P_1 + P_2) = \mathrm{Infl}(P_1) + \mathrm{Infl}(P_2)$ for all P_1 and P_2.

Nonconstant positivity comes from the observation that any nonconstant component has nonzero influence on a linear regression model, and the other two axioms are reasonable properties for

an influence measure. Unfortunately, these axioms are inconsistent. To see this, let P be a nonconstant random variable. Then, $\mathrm{Infl}(P) + \mathrm{Infl}(-P) > 0$ by nonconstant positivity, but by additivity, $\mathrm{Infl}(P) + \mathrm{Infl}(-P) = \mathrm{Infl}(P - P) = \mathrm{Infl}(0) = 0$. The issue is that two random variables can cancel out, so we relax the additivity axiom so that it only applies when the two variables are independent:

Independent additivity. If P_1 and P_2 are independent, $\mathrm{Infl}(P_1 + P_2) = \mathrm{Infl}(P_1) + \mathrm{Infl}(P_2)$.

Because linear models cannot capture any nonlinear associations, two variables with zero covariance may as well be independent as far as the linear model is concerned. This fact motivates the final version of the additivity axiom, shown below:

Zero-covariance additivity. If $\mathrm{Cov}(P_1, P_2) = 0$, $\mathrm{Infl}(P_1 + P_2) = \mathrm{Infl}(P_1) + \mathrm{Infl}(P_2)$.

Now I prove that the combination of nonnegativity, nonconstant positivity, and zero-covariance additivity require the use of variance as the influence measure. I first argue that $\mathrm{Infl}(P) + \mathrm{Infl}(-P)$ must be proportional to the variance of P, and then I proceed to show that $\mathrm{Infl}(P)$ and $\mathrm{Infl}(-P)$ must in fact be equal.

Lemma 2.1. *Suppose* $\mathrm{Var}(P_1) = \mathrm{Var}(P_2)$. *If* Infl *satisfies nonnegativity, nonconstant positivity, and zero-covariance additivity, then* $\mathrm{Infl}(P_1) + \mathrm{Infl}(-P_1) = \mathrm{Infl}(P_2) + \mathrm{Infl}(-P_2)$.

Proof. Let P be an independent random variable such that $\mathrm{Var}(P) = \mathrm{Var}(P_1) = \mathrm{Var}(P_2)$. It is always possible to find such P; for example, P could be i.i.d. with P_1. Because $\mathrm{Cov}(P+P_1, P-P_1) = \mathrm{Var}(P) - \mathrm{Var}(P_1) = 0$, by zero-covariance additivity we have

$$\mathrm{Infl}(2P) = \mathrm{Infl}(P + P_1) + \mathrm{Infl}(P - P_1) = \mathrm{Infl}(P) + \mathrm{Infl}(P_1) + \mathrm{Infl}(P) + \mathrm{Infl}(-P_1).$$

Similarly, $\mathrm{Infl}(2P) = \mathrm{Infl}(P) + \mathrm{Infl}(P_2) + \mathrm{Infl}(P) + \mathrm{Infl}(-P_2)$, and our desired result follows from combining the above two equations. □

Theorem 2.2. *Let* Infl *be an influence measure that satisfies nonnegativity, nonconstant positivity, and zero-covariance additivity. Then,* $\mathrm{Infl}(P) + \mathrm{Infl}(-P) = c\,\mathrm{Var}(P)$*, where* c *can be any fixed positive constant.*

Proof. By Lemma 2.1, $\mathrm{Infl}(P) + \mathrm{Infl}(-P) = f(\mathrm{Var}(P))$ for some function f with domain $[0, \infty)$. It remains to show that $f(x)$ must have the form cx for some positive constant c.

Let P_1 and P_2 be independent random variables with variances v_1 and v_2, respectively. Then, $\mathrm{Var}(P_1 + P_2) = v_1 + v_2$, and by zero-covariance additivity, we have

$$\begin{aligned} f(v_1 + v_2) &= \mathrm{Infl}(P_1 + P_2) + \mathrm{Infl}(-P_1 - P_2) \\ &= \mathrm{Infl}(P_1) + \mathrm{Infl}(-P_1) + \mathrm{Infl}(P_2) + \mathrm{Infl}(-P_2) = f(v_1) + f(v_2) \end{aligned}$$

for all nonnegative v_1 and v_2. This is only possible if f is a linear function of x. Therefore, we can write $f(x) = cx$ for some c, which must be positive due to nonconstant positivity. □

Having shown that $\mathrm{Infl}(P) + \mathrm{Infl}(-P)$ is a linear function of $\mathrm{Var}(P)$, we now we want to show that $\mathrm{Infl}(P) = \mathrm{Infl}(-P)$. For brevity, I will write $\gamma(P) = \frac{1}{2}(\mathrm{Infl}(P)+\mathrm{Infl}(-P))$ to denote what $\mathrm{Infl}(P)$ would be if it is indeed true that $\mathrm{Infl}(P) = \mathrm{Infl}(-P)$. Now consider $\mathrm{Infl}(P)$ as the

sum of two expressions $\gamma(P)$ and $\mathrm{Infl}(P) - \gamma(P)$. Lemma 2.3 shows that these two expressions have different growth rates in P.

Lemma 2.3. *Let* $\gamma(P) = \frac{1}{2}(\mathrm{Infl}(P) + \mathrm{Infl}(-P))$. *If* Infl *satisfies nonnegativity, nonconstant positivity, and zero-covariance additivity, then* $\mathrm{Infl}(2^n P) = 2^n(\mathrm{Infl}(P) - \gamma(P)) + 4^n\gamma(P)$ *for all integer* n.

Proof. First, I prove the theorem for all nonnegative n. The proof is by induction on n, and the base case where $n = 0$ is trivial.

Let P' be an independent random variable such that $\mathrm{Var}(P) = \mathrm{Var}(P')$. Then, $\mathrm{Cov}(P + P', P - P') = \mathrm{Var}(P) - \mathrm{Var}(P') = 0$, so by zero-covariance additivity,

$$\begin{aligned}\mathrm{Infl}(2^{n+1}P) &= \mathrm{Infl}(2^n(P + P')) + \mathrm{Infl}(2^n(P - P')) \\ &= \mathrm{Infl}(2^n P) + \mathrm{Infl}(2^n P') + \mathrm{Infl}(2^n P) + \mathrm{Infl}(-2^n P') \\ &= 2\,\mathrm{Infl}(2^n P) + (\mathrm{Infl}(2^n P') + \mathrm{Infl}(-2^n P')).\end{aligned}$$

Using the inductive hypothesis, we can turn the right-hand side into

$$(2^{n+1}(\mathrm{Infl}(P) - \gamma(P)) + 2 \cdot 4^n\gamma(P)) + 2\,\gamma(2^n P').$$

Finally, because γ is proportional to variance, we can substitute in $\gamma(2^n P') = \gamma(2^n P) = 4^n\gamma(P)$ to get

$$2^{n+1}(\mathrm{Infl}(P) - \gamma(P)) + 4^{n+1}\gamma(P),$$

which is what we want.

Now, consider the case where n is negative. Then, because $-n$ is positive, we have $\mathrm{Infl}(2^{-n}P) = 2^{-n}(\mathrm{Infl}(P) - \gamma(P)) + 4^{-n}\gamma(P)$ for all P. Substituting $2^n P$ for P and noting that $\gamma(2^n P) = 4^n\gamma(P)$, we get

$$\mathrm{Infl}(P) = 2^{-n}\mathrm{Infl}(2^n P) - 2^n\gamma(P) + \gamma(P),$$

and a bit of algebraic manipulation gives us the desired result. □

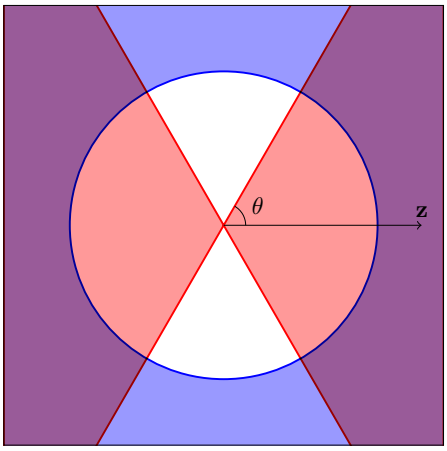
Lemma 2.3 says that $\gamma(P)$ scales quadratically with P, whereas $\mathrm{Infl}(P) - \gamma(P)$ is only linear in P. Because of this discrepancy in scaling, if $\mathrm{Infl}(P)$ differs from $\gamma(P)$, we can repeatedly halve P to end up with a random variable whose influence is negative. But this contradicts the nonnegativity axiom, so we must have $\mathrm{Infl}(P) = \gamma(P)$ for all P. This informal argument is formalized below.

Theorem 2.4. *Let* Infl *be an influence measure that satisfies nonnegativity, nonconstant positivity, and zero-covariance additivity. Then,* $\mathrm{Infl}(P) = \gamma(P)$ *for all* P.

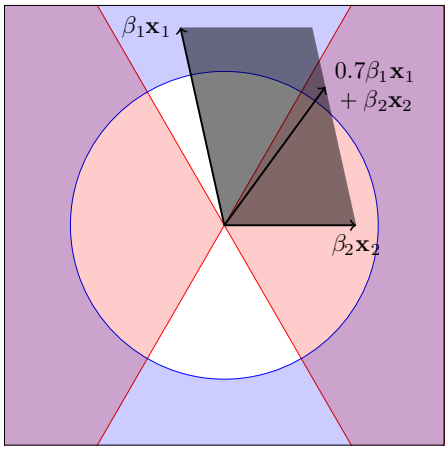
Proof. Suppose for contradiction that $\mathrm{Infl}(P) \neq \gamma(P)$. If $\mathrm{Infl}(P) > \gamma(P)$, then $\mathrm{Infl}(-P) < \gamma(-P)$, so we can assume without loss of generality that $\gamma(P) - \mathrm{Infl}(P) > 0$. Choose a small enough n such that $2^n < \frac{\gamma(P) - \mathrm{Infl}(P)}{\gamma(P)}$. Then, by Lemma 2.3,

$$\mathrm{Infl}(2^n P) = 2^n(\mathrm{Infl}(P) - \gamma(P)) + 4^n\gamma(P) = 2^n\gamma(P)\left(2^n - \frac{\gamma(P) - \mathrm{Infl}(P)}{\gamma(P)}\right) < 0,$$

which contradicts the nonnegativity axiom. □



(a) $\mathbf{z}$ is a vector representation of the protected attribute Z, and components of the model can also be represented as vectors. If a component is inside the red double cone, it exceeds the association threshold ϵ, where the angle θ is set such that $\epsilon = \cos^2\theta$. The cone on the right side corresponds to positive correlation with Z, and the left cone negative correlation. Components in the blue shaded area exceed some influence threshold δ. If any component exceeds both the association and the influence thresholds, it is a proxy and may be disallowed.



(b) $\mathbf{x}_1$ and $\mathbf{x}_2$ are vector representations of X_1 and X_2, which are inputs to the model $\hat{Y} = \beta_1 X_1 + \beta_2 X_2$. The gray shaded area indicates the space of all possible components of the model. $\beta_1 X_1$ is a component, but it is not a proxy because it does not have strong enough association with Z. Although $\beta_2 X_2$ is strongly associated with Z, it is not influential enough to be a proxy. On the other hand, $0.7\beta_1 X_1 + \beta_2 X_2$ is a component that exceeds both the association and the influence thresholds, so it is a proxy and may be disallowed.

Figure 2.1: Illustration of proxy use with the vector interpretation of random variables. In the above examples, all vectors lie in $\mathbb{R}^2$ for ease of depiction. In general, the vectors $\mathbf{z}, \mathbf{x}_1, \ldots, \mathbf{x}_d$ can span $\mathbb{R}^{d+1}$.

Finally, Theorem 2.5 states that the influence measure must be proportional to variance. This result follows from unpacking the definition of $\gamma(P)$, so the proof is omitted. Definition 2.3 simply chooses the constant of proportionality that gives unit influence to the whole model.

Theorem 2.5. *If* Infl *satisfies nonnegativity, nonconstant positivity, and zero-covariance additivity, then* $\mathrm{Infl}(P) \propto \mathrm{Var}(P)$.

2.2.3 Proxy Detection Algorithm

I now present the proxy detection algorithms, which take advantage of properties specific to linear regression to quickly identify components of interest. I prove that we can use an exact optimization problem (Problem 2.1) to either identify a proxy if one exists, or definitively conclude that there is no proxy. However, because this problem is not convex and in some cases may be intractable, I also present an approximate version of the problem (Problem 2.2) that sacrifices some precision. The approximate algorithm can still be used to conclude that a model does not have any proxies, but it may return false positives. In Section 2.2.4, I evaluate how these algorithms

perform on real-world data.

Covariance as Inner Product

Because the only operations that we perform on random variables are addition and scalar multiplication, we can safely treat the random variables as vectors in a vector space. In addition, covariance is an inner product in this vector space. As a result, it is helpful to think of random variables $Z, X_1, \ldots, X_d$ as vectors $\mathbf{z}, \mathbf{x}_1, \ldots, \mathbf{x}_d \in \mathbb{R}^{d+1}$, with covariance as dot product. Under this interpretation, influence is characterized by $\mathrm{Infl}_{\hat{Y}}(P) \propto \mathrm{Var}(P) = \mathrm{Cov}(P, P) = \mathbf{p} \cdot \mathbf{p} = \|\mathbf{p}\|^2$, where $\|\cdot\|$ denotes the L^2-norm, and association is shown in Eq. (2.3), where θ is the angle between the two vectors $\mathbf{p}$ and $\mathbf{z}$.

$$\mathrm{Asc}(P, Z) = \frac{\mathrm{Cov}(P, Z)^2}{\mathrm{Var}(P)\mathrm{Var}(Z)} = \left(\frac{\mathbf{p} \cdot \mathbf{z}}{\|\mathbf{p}\|\|\mathbf{z}\|}\right)^2 = \cos^2\theta, \tag{2.3}$$

This abstraction is illustrated in more detail in Fig. 2.1.

To find coordinates for the vectors, consider the covariance matrix $[\mathrm{Cov}(X_i, X_j)]_{i,j \in \{0,\ldots,n\}}$, where $Z = X_0$ for notational convenience. If we can write this covariance matrix as $A^T A$ for some $(d{+}1) \times (d{+}1)$ matrix A, then each entry in the covariance matrix is the dot product of two (not necessarily distinct) columns of A. In other words, the mapping from the random variables $Z, X_1, \ldots, X_d$ to the columns of A preserves the inner product relationship. Now it remains to decompose the covariance matrix into the form $A^T A$. Since the covariance matrix is guaranteed to be positive semidefinite, two of the possible decompositions are the Cholesky decomposition and the matrix square root.

Proxy Detection Algorithm

The proxy detection algorithms use as subroutines the optimization problems that are formally stated in Fig. 2.2. I first motivate the exact optimization problem (Problem 2.1) and show how the solutions to these problems can be used to determine whether the model contains a proxy. Then, I present the approximate optimization problem (Problem 2.2), which sacrifices exactness for efficient solvability.

Let A' be the $(d{+}1) \times d$ matrix $\begin{bmatrix}\beta_1 \mathbf{x}_1 & \ldots & \beta_d \mathbf{x}_d\end{bmatrix}$. The constraint $0 \preceq \boldsymbol{\alpha} \preceq 1$ restricts the solutions to be inside the space of all components, represented by the gray shaded area in Fig. 2.1b. Moreover, when $s \in \{-1, 1\}$, the constraint $\|A'\boldsymbol{\alpha}\| \leq s \cdot (\mathbf{z}^T A'\boldsymbol{\alpha})/(\sqrt{\epsilon}\|\mathbf{z}\|)$ describes one of the red cones in Fig. 2.1a, which together represent the association constraint. Subject to these constraints, we maximize the influence, which is proportional to $\|A'\boldsymbol{\alpha}\|^2$. Theorem 2.6 shows that this technique is sufficient to determine whether a model contains a proxy.

Theorem 2.6. *Let P denote the component defined by the alpha-coefficients $\boldsymbol{\alpha}$. The linear regression model $\hat{Y} = \beta_1 X_1 + \cdots + \beta_d X_d$ contains a proxy if and only if there exists a solution to Problem 2.1 with $s \in \{-1, 1\}$ such that $\mathrm{Infl}_{\hat{Y}}(P) \geq \delta$.*

Proof. It is clear that P is a component if and only if the constraint $0 \preceq \boldsymbol{\alpha} \preceq 1$ is satisfied. I now show that the second constraint in Problem 2.1 is the correct association constraint.

Problem 2.1 Exact optimization

$$\begin{array}{ll} \min & -\|A'\boldsymbol{\alpha}\|^2 \\ \text{s.t.} & 0 \preceq \boldsymbol{\alpha} \preceq 1 \text{ and } \|A'\boldsymbol{\alpha}\| \leq s \cdot \dfrac{\mathbf{z}^T A'\boldsymbol{\alpha}}{\sqrt{\epsilon}\|\mathbf{z}\|} \end{array}$$

Problem 2.2 Approximate optimization

$$\begin{array}{ll} \min & -\mathbf{c}^T\boldsymbol{\alpha} \\ \text{s.t.} & 0 \preceq \boldsymbol{\alpha} \preceq 1 \text{ and } \|A'\boldsymbol{\alpha}\| \leq s \cdot \dfrac{\mathbf{z}^T A'\boldsymbol{\alpha}}{\sqrt{\epsilon}\|\mathbf{z}\|} \end{array}$$

Figure 2.2: Optimization problems used to find proxies in linear regression models. A' is the $(d{+}1) \times d$ matrix $\begin{bmatrix}\beta_1\mathbf{x}_1 & \dots & \beta_d\mathbf{x}_d\end{bmatrix}$, and we optimize over $\boldsymbol{\alpha}$, which is an d-dimensional vector of the alpha-coefficients used in Definition 2.1. ϵ is the association threshold, and $\mathbf{c}$ is the d-dimensional vector that satisfies $c_i = \|\beta_i\mathbf{x}_i\|$.

Let $\mathbf{p} = \alpha_1\beta_1\mathbf{x}_1 + \cdots + \alpha_d\beta_d\mathbf{x}_d = A'\boldsymbol{\alpha}$ be the vector representation of P. From Eq. (2.3), we know that $\mathrm{Asc}(P, Z) \geq \epsilon$ if and only if $(\mathbf{p} \cdot \mathbf{z})^2 \geq \epsilon(\|\mathbf{p}\|\|\mathbf{z}\|)^2$. This inequality holds if and only if

$$\frac{\mathbf{p} \cdot \mathbf{z}}{\sqrt{\epsilon}\|\mathbf{z}\|} \geq \|\mathbf{p}\| \text{ or } -\frac{\mathbf{p} \cdot \mathbf{z}}{\sqrt{\epsilon}\|\mathbf{z}\|} \geq \|\mathbf{p}\|,$$

where the former inequality represents positive correlation between P and Z, and the latter negative correlation. Making the substitution $\mathbf{p} = A'\boldsymbol{\alpha}$, we see that P is a component with strong enough association if and only if it meets the constraints in Problem 2.1 with either $s = 1$ or $s = -1$.

Therefore, if P is a solution to the optimization problem and $\mathrm{Infl}_{\hat{Y}}(P) \geq \delta$, it is a component that exceeds both the association and influence thresholds, which means that it is a proxy. This proves the reverse direction of the theorem statement.

To prove the forward direction, we consider the influence. We have $\mathrm{Infl}_{\hat{Y}}(P) \propto \mathrm{Var}(P) = \|\mathbf{p}\|^2 = \|A'\boldsymbol{\alpha}\|^2$, so minimizing $-\|A'\boldsymbol{\alpha}\|^2$, as done in Problem 2.1, has the effect of maximizing the influence.

Suppose that P is a proxy, i.e., P is a component such that $\mathrm{Asc}(P, Z) \geq \epsilon$ and $\mathrm{Infl}_{\hat{Y}}(P) \geq \delta$. Without loss of generality, $\mathrm{Cov}(P, Z) > 0$. Then, P satisfies the constraints of Problem 2.1 with $s = 1$, so the solution to the optimization problem must be at least as influential as P. The forward direction of the theorem statement follows from the assumption that P has influence at least δ. □

In essence, Theorem 2.6 guarantees the correctness of the following proxy detection algorithm: Run Problem 2.1 with $s = 1$ and $s = -1$, and compute the association and influence of the resulting solutions. The model contains a proxy if and only if any of the solutions passes both the association and the influence thresholds.

It is worth mentioning that Problem 2.1 tests for strong positive correlation with Z when $s = 1$ and for strong negative correlation when $s = -1$. This optimization problem resembles a second-order cone program (SOCP) [14, §4.4.2], which can be solved efficiently. However, the objective function is concave, so the standard techniques for solving SOCPs do not work on this problem. To get around this issue, we can instead solve Problem 2.2, which has a linear objective function whose coefficients $c_i = \|\beta_i\mathbf{x}_i\|$ were chosen so that the inequality $\|A'\boldsymbol{\alpha}\| \leq \mathbf{c}^T\boldsymbol{\alpha}$ always

holds. This inequality allows us to prove Theorem 2.7, which mirrors the claim of Theorem 2.6 but only in one direction.

Theorem 2.7. *If the linear regression model $\hat{Y} = \beta_1 X_1 + \cdots + \beta_d X_d$ contains a proxy, then there exists a solution to Problem 2.2 with $s \in \{-1, 1\}$ such that $\mathbf{c}^T \boldsymbol{\alpha} \geq (\delta \operatorname{Var}(\hat{Y}))^{0.5}$.*

Proof. Because the proof is very similar to the proof of the forward direction of Theorem 2.6, I only point out where the proofs differ. Let P be a proxy of $\hat{Y}$, and let $\boldsymbol{\alpha}$ be the corresponding vector of alpha-coefficients. I will show that $\mathbf{c}^T \boldsymbol{\alpha} \geq (\delta \operatorname{Var}(\hat{Y}))^{0.5}$.

By the definition of proxy, we have $\operatorname{Infl}_{\hat{Y}}(P) = \operatorname{Var}(P)/\operatorname{Var}(\hat{Y}) \geq \delta$. Since $\operatorname{Var}(P) = \|\mathbf{p}\|^2 = \|A'\boldsymbol{\alpha}\|^2$, we can rewrite this as $\|A'\boldsymbol{\alpha}\| \geq (\delta \operatorname{Var}(\hat{Y}))^{0.5}$. Finally, the desired inequality follows from the observation that $\|A'\boldsymbol{\alpha}\| = \|\sum_{i=1}^{d} \alpha_i \beta_i \mathbf{x}_i\| \leq \sum_{i=1}^{d} \alpha_i \|\beta_i \mathbf{x}_i\| = \mathbf{c}^T \boldsymbol{\alpha}$ by the triangle inequality. □

Theorem 2.7 suggests a quick algorithm to verify that a model does not have any proxies. We solve the SOCP described by Problem 2.2, once with $s = 1$ and once with $s = -1$. If neither solution satisfies $\mathbf{c}^T \boldsymbol{\alpha} \geq (\delta \operatorname{Var}(\hat{Y}))^{0.5}$, by the contrapositive of Theorem 2.7, we can be sure that the model does not contain any proxies.

However, the converse does not hold, i.e., we cannot be sure that the model has a proxy even if a solution to Problem 2.2 satisfies $\mathbf{c}^T \boldsymbol{\alpha} \geq (\delta \operatorname{Var}(\hat{Y}))^{0.5}$. This is because $\mathbf{c}^T \boldsymbol{\alpha}$ overapproximates $\|A'\boldsymbol{\alpha}\|$ by using the triangle inequality. As a result, it is possible for the influence to be below the threshold even if the value of $\mathbf{c}^T \boldsymbol{\alpha}$ is above the threshold. While there is in general no upper bound on the overapproximation factor of the triangle inequality, the experiments in Section 2.2.4 show that this factor is not too large in practice. In addition, Problem 2.1 often works well enough in practice despite not being a convex optimization problem.

2.2.4 Empirical Evaluation

In this section, I evaluate the performance of Problems 2.1 and 2.2 on real-world predictive policing datasets. I ran the proxy detection algorithms on observational data from Chicago's Strategic Subject List (SSL) model [24] and the Communities and Crimes (C&C) dataset [33]. The creator of the SSL model claims that the model avoids variables that could lead to discrimination [9], and if this is the case then we would expect to see only weak proxies if any. On the other hand, the C&C dataset contains many variables that are correlated with race, so we would expect to find strong proxies in a model trained with this dataset. To test these hypotheses, I implemented Problems 2.1 and 2.2 with the `cvxopt` package [4] in Python. The experimental results confirm the hypotheses and show that the algorithms run very quickly (< 1 second).

For each dataset, I briefly describe the dataset and present the experimental results, demonstrating how the identified proxies can provide evidence of discriminatory behavior in models. Then, I explain the implications of these results on the false positive and false negative rates in practice, and I discuss how a practitioner can decide which values of ϵ and δ to use.

Association threshold ϵ	0.01	0.02	0.03	0.04	0.05	0.06
Actual infl. (Problem 2.1)	0.8816	0.2263	0.1090	0.0427*	0.0065*	0.0028
Approx. infl. (Problem 2.2)	1.6933	0.6683	0.3820	0.1432	0.0270	0.0085
Actual infl. (Problem 2.2)	0.8476	0.1874	0.0987	0.0420	0.0080	0.0027

Table 2.1: Influence of the components obtained by solving the exact (Problem 2.1) and approximate (Problem 2.2) optimization problems for the SSL model using $Z = \text{race}$ and $s = 1$. No component had strong enough association when $s = -1$ instead. Asterisks indicate that the exact optimization problem terminated early due to a singular KKT matrix. The approximate optimization problem did not have this issue, and the overapproximation that it makes of the components' influence is shown in the second row.

Strategic Subject List

The SSL [24] is a model that the Chicago Police Department uses to assess an individual's risk of being involved in a shooting incident, either as a victim or a perpetrator. The SSL dataset consists of 398,684 rows, each of which corresponds to a person. Each row includes the SSL model's eight input variables (including age, number of previous arrests for violent offenses, and whether the person is a member of a gang), the SSL score given by the model, and the person's race and gender.

I searched for proxies for race (binary Black/White) and gender (binary male/female), filtering out rows with other race or gender. After also filtering out rows with missing data, we were left with 290,085 rows. Because I did not have direct access to the SSL model, I trained a linear regression model to predict the SSL score of a person given the same set of variables that the SSL model uses. This new model explains approximately 80% of the variance in the SSL scores, so it is a reasonable approximation of the true model for the purposes of this evaluation.

The strengths of the proxies for race are given in Table 2.1. The estimated influence was computed as $(\mathbf{c}^T\boldsymbol{\alpha})^2/\mathrm{Var}(\hat{Y})$, which is the result of solving for δ in the inequality given in Theorem 2.7. I found that this estimate is generally about 3–4$\times$ larger than the actual influence. Although the proxies for race were somewhat stronger than those for gender, neither type had significant influence ($\delta > 0.05$) beyond small ϵ levels (≈ 0.03–0.04). This is consistent with the aforementioned hypothesis about the lack of discriminatory behavior in this model.

Communities and Crimes

C&C [82] is a dataset in the UCI machine learning repository [33] that combines socioeconomic data from the 1990 US census with the 1995 FBI Uniform Crime Reporting data. It consists of 1,994 rows, each of which corresponds to a community (e.g., municipality) in the U.S., and 122 potential input variables. After I removed the variables that directly measure race and the ones with missing data, there were 90 input variables remaining.

I simulated a hypothetical naive attempt at predictive policing by using this dataset to train a linear regression model that predicts the per capita rate of violent crimes in a community. The protected attribute Z was defined as the difference between the percentages of people in the community who are Black and White, respectively. There is a strong association in the dataset

between the rate of violent crime and Z ($\mathrm{Asc}(Y, Z) = 0.48$), and the model amplifies this bias even more ($\mathrm{Asc}(\hat{Y}, Z) = 0.65$).

As expected, I found very strong proxies for race in the model trained with the C&C dataset. For example, one proxy consisting of 58 of the 90 input variables achieves an influence of 0.34 when $\epsilon = 0.85$. Notably, the input variable most strongly associated with race has an association of only 0.73, showing that *in practice multiple variables combine to result in a stronger proxy than any of the individual variables.* In addition, the model contains a proxy whose association is 0.40 and influence is 14.5. In other words, the variance of the proxy is *14.5 times greater than that of the model*; this arises because other associated variables cancel most of this variance in the full model.

False Positives and False Negatives

Theorem 2.6 shows that the exact proxy detection algorithm detects a proxy if and only if the model in fact contains a proxy. In other words, if Problem 2.1 returns optimal solutions, we can use the solutions to conclusively determine whether there exists a proxy, and there will be no false positives or false negatives. However, sometimes Problem 2.1 terminates early due to a singular KKT matrix, and in this case one can turn to the approximate proxy detection algorithm.

Although Problem 2.2 sometimes returns solutions that are not in fact proxies, we can easily ascertain whether any given solution is a proxy by simply computing its association and influence. However, even if the solution returned by Problem 2.2 turn out to not be proxies, the model could still contain a different proxy. Using Table 2.1 as reference, we see that this happens in the SSL model if, for example, $\epsilon = 0.02$ and δ is between 0.1874 and 0.2263. Therefore, one can consider the approximate algorithm as giving a finding of either "potential proxy use" or "no proxy use". Theorem 2.7 shows that a finding of "no proxy use" does indeed guarantee that the model is free of proxies. In other words, the approximate algorithm has no false negatives. However, the algorithm overapproximates influence, so the algorithm can give a finding of "potential proxy use" when there are no proxies, resulting in a false positive. This happens when δ is between the maximum feasible influence (first row in Table 2.1) and the maximum feasible overapproximation of influence (second row in Table 2.1).

Reasonable Values of ϵ and δ

Although the appropriate values of ϵ and δ depend on the application, I would like to remind the reader that association is the *square* of the Pearson correlation coefficient. This means that an association of 0.05 corresponds to a Pearson correlation coefficient close to 0.22, which represents a not-insignificant amount of correlation. Likewise, influence is proportional to variance, which increases quadratically with scalar coefficients. Therefore, I recommend against setting ϵ and δ to a value much higher than 0.05. To get an idea of which values of δ are suitable for a particular application, the practitioner can compare the proposed value of δ against the influence of the individual input variables $\beta_i X_i$.

2.3 Approximate, Model-Agnostic Detection of Proxy Use

When the model to be audited is not a linear regression model, the methods presented in the previous section are not applicable. Moreover, unlike the linear regression setting, the general setting preserves a meaningful distinction between black-box and white-box audits, so black-box auditing methods cannot assume much about the structure of the model. In this section, I describe FlipTest [13], a black-box auditing procedure that uncovers evidence of proxy-like behavior in this general setting. As the only assumption of FlipTest is that the model is a binary classifier (i.e., $\hat{Y} \in \{0, 1\}$), it has the advantage that does not require cooperation from the model holder.

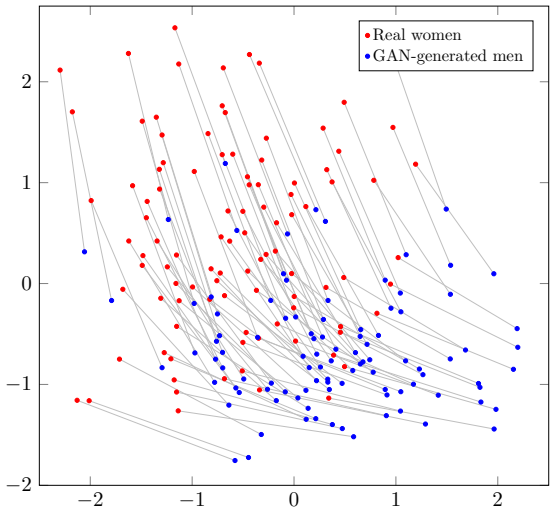
2.3.1 Example of FlipTest Analysis

I will illustrate the main concepts behind FlipTest with a running example, which uses a synthetic dataset created by Lipton et al. [65, §4.1]. This dataset consists of two features, hair length and work experience, and supposes a binary classifier that uses these features to decide whether a given person should be hired. We search for possible gender bias in this model, asking whether the model's output would have been different had a given person been of a different gender. However, it is not sufficient to simply flip the gender attribute due to correlations in the data that the model may use as a proxy for gender: in this data, gender is correlated with hair length and work experience. Additionally, flipping the gender attribute is not an option because the model does not directly use this attribute.

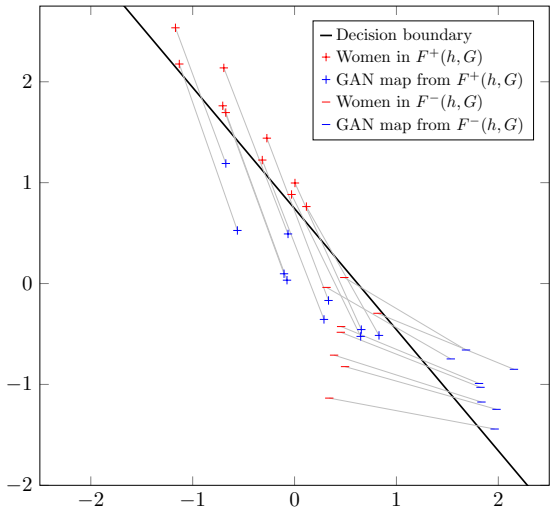
FlipTest instead maps the set of women in the data to the set of men in the data and analyzes the cases where the model treats women differently from the men that they are mapped to. This raises the question of which specific man a given woman should be mapped to. FlipTest's answer to this question is based on the intuition that a difference in treatment between two people is not by itself strong evidence of discrimination unless they are similar enough that the disparity cannot be justified. For example, when a man with 20 years of relevant experience is hired over a woman with no experience, this difference would likely be attributed to work experience rather than gender discrimination. This motivates the use of an *optimal transport mapping* [100], which minimizes the sum of the distances between a woman and the man that she is mapped to (her *counterpart*). Here, the distance quantifies how different a pair of people are.

It remains to specify a distance function, which is typically called the *cost function* in the context of optimal transport. Although there are no easy answers to the question of which people are similar for the purpose of establishing discrimination, the goal of FlipTest is to demonstrate a new technique for finding evidence of possible discrimination rather than to present a conclusive definition of discrimination. Here and in the rest of this chapter, I use the square of the L^1 distance, but FlipTest is compatible with any cost function deemed suitable for the setting.

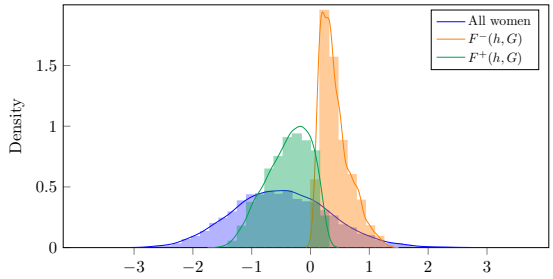
We then analyze the optimal transport mapping, which is depicted in Fig. 2.3a. The analysis is through the *flipset* $F(h, G)$, which consists of all women whose outcomes were different from their counterparts'. We partition the flipset into the *positive flipset* $F^+(h, G)$, which contains the hired women whose counterparts were not, and the *negative flipset* $F^-(h, G)$, which is the set of rejected women whose counterparts were hired. Thus, in some sense the women in $F^+(h, G)$ were advantaged due to their gender, and those in $F^-(h, G)$ disadvantaged. Although this is not sufficient to establish that gender *caused* the difference in the way that some causal tests [29, 61]



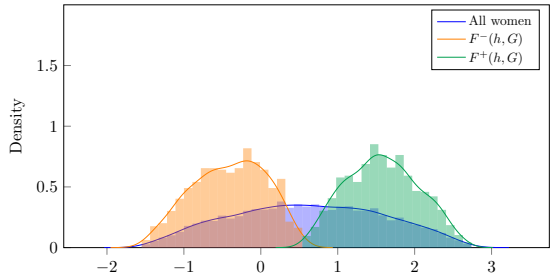
(a) Optimal transport mapping generated by a GAN approximation. The horizontal axis represents normalized work experience, and the vertical axis represents normalized hair length.



(b) Flipsets as defined by a model with the given decision boundary. The horizontal axis represents normalized work experience, and the vertical axis represents normalized hair length.



(c) Marginal distribution of work experience for all women, negative flipset, and positive flipset.



(d) Marginal distribution of work experience for all women, negative flipset, and positive flipset.

Figure 2.3: Demonstration of FlipTest on a synthetic dataset created by Lipton et al. [65].

can, FlipTest has the advantage that it queries the model on in-distribution points only, so its response to these inputs is likely to be reliable.

We begin by examining the size of the positive and negative flipsets. Suppose that the model does not satisfy statistical parity, hiring disproportionately more men than women. Then, the flipsets will have different sizes, with the negative flipset larger than the positive. Therefore, a large difference in the sizes of the flipsets is evidence of possibly discriminatory behavior in the model. However, such differences may also be based on a justifiable reason, such as when the job in question has requirements that are more likely to be satisfied by a particular gender.

Alternatively, as is the case here, the positive and negative flipsets could have the same size. Then, the *net* flipset size is zero, and the model satisfies statistical parity. However, if the sizes of the positive and negative flipsets are still large, then there may be discrimination at the subgroup level. To investigate which subgroups may be discriminated against (or unfairly advantaged), we can compare the distributions of the flipsets to that of the entire population. Figs. 2.3c and 2.3d plot the marginal distributions, which show that the advantaged ($F^+(h, G)$) women tend to have much longer hair than the disadvantaged ($F^-(h, G)$) women. This suggests that the model may be discriminating against short-haired women.

This result is consistent with previous findings by Lipton et al. [65]. They created this synthetic dataset to argue that some fair learning algorithms employ a "problematic within-class discrimination mechanism". In particular, they show that the model achieves statistical parity by using hair length as a proxy for gender. Thus, in order to hire more women, the model rewards long hair at the expense of short-haired women, and FlipTest correctly detects this behavior.

Finally, a *transparency report* gives more information about why the model may behave in this way. The transparency report describes how the members of the flipsets are different from their counterparts, shedding light on which features may have contributed to the model's decision to classify the counterparts differently. As we can see in Fig. 2.3b, the women in the negative flipset have much less work experience than their counterparts, and this suggests that they were not hired because of inadequate work experience. In other words, work experience appears to be a proxy for gender.

However, this method is not foolproof because work experience could have been a correlate of another feature that the model actually uses. Still, it identifies an area for further investigation that can determine which feature is the most causally responsible for the model's behavior on this subgroup. In Section 2.4, I supplement FlipTest with QII [29], which is a black-box method of causal analysis. Although there are many causal analysis methods, including the original proxy detection algorithm by Datta et al. [30], QII has the advantage that it keeps the entire auditing pipeline black-box, making the audit possible even when the model holder is not cooperative.

If the causal analysis reveals that work experience causes the difference in the hiring decisions—and in our example it does—a practitioner can consult a domain expert to decide whether the use of work experience is justified. In many cases it would be justified, but it may not if the disparity in work experience is due to discrimination.

2.3.2 Optimal Transport Mapping

In this section, I describe the optimal transport problem in more detail, and show how to solve a GAN objective to approximate the optimal transport mapping in a way that generalizes to unseen

data points. I then compare this GAN approximation to the exact optimal transport mapping and another approximation method by Seguy et al. [84], finding that the GAN tends to give the most stable mappings. Although this GAN approximation is not the only way to operationalize FlipTest, we use the GAN approximation throughout the rest of Section 2.3 because it tends to give more reliable results than the alternatives.

I now introduce the notation. Let $\mathcal{S}$ and $\mathcal{S}'$ be two distributions defined over the feature space $\mathcal{X}$. In practice, we do not know these distributions, so we usually deal with observations of points drawn from these distributions instead. We will use the sets $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and $S' = \{\mathbf{x}'_1, \ldots, \mathbf{x}'_n\}$ to denote the observed points, where $n = |S| = |S'|$. Note that here we assume that $|S| = |S'|$ for ease of exposition, as this assumption allows the resulting exact optimal transport mapping to be deterministic. The general case where $|S| \neq |S'|$ can be handled through the use of randomized optimal transport mappings, and the GAN approximation method does not require that the two sets have equal size.

Let $c : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ be a cost function that describes the cost of moving between two points in the feature space $\mathcal{X}$. Intuitively, an optimal transport mapping from S to S' is a minimum-cost way to move the points in S such that the end result is S'. Thus, if more similar pairs of points have a lower cost, an optimal transport mapping describes how to match points in S with their similar counterparts in S'. Formally, an optimal transport mapping can be defined as a bijection $f : S \to S'$ that minimizes the expected cost $\mathbb{E}[c(\mathbf{x}, f(\mathbf{x}))] = \frac{1}{n}\sum_{i=1}^{n} c(\mathbf{x}_i, f(\mathbf{x}_i))$.

Approximation via GANs

While the exact optimal transport mapping between S and S' can be solved through a linear program or the Hungarian algorithm [60], these methods do not scale well to large n. In addition, the exact mapping, as well as some approximations thereof [3, 78], is not defined for points outside of S. Therefore, I instead use a generative adversarial network (GAN) [39] to approximate an optimal transport mapping in a way that avoids both of these issues.

For concreteness, I will describe a construction based on the Wasserstein GAN [7], but Theorem 2.8 can be extended to other types of GANs as well. Because we want to use the generator G of the GAN as an optimal transport mapping, its inputs will draw randomness from $\mathcal{S}$. When training a conventional Wasserstein GAN with the sets of observed points S and S', the generator's loss function is $(1/n)\sum_{\mathbf{x}\in S} D(G(\mathbf{x}))$ for discriminator D, and the discriminator's loss function is $(1/n)\sum_{\mathbf{x}'\in S'} D(\mathbf{x}') - (1/n)\sum_{\mathbf{x}\in S} D(G(\mathbf{x}))$. For the purpose of finding an optimal transport mapping, we modify the generator's loss function to take into account the cost of moving from a point in S to a point in S':

$$L_G = \frac{1}{n}\sum_{\mathbf{x}\in S} D(G(\mathbf{x})) + \frac{\lambda}{n}\sum_{\mathbf{x}\in S} c(\mathbf{x}, G(\mathbf{x})) \tag{2.4}$$

The modified generator has two objectives, with the parameter λ controlling their relative importance: generating the correct output distribution $\mathcal{S}'$, and minimizing the expected cost $c(\mathbf{x}, G(\mathbf{x}))$. Theorem 2.8 formalizes the intuition that these objectives are also those of an optimal transport mapping.

Theorem 2.8. *Suppose that G^* is a minimizer of L_G among all G such that $G(S) = S'$. If $\lambda > 0$, G^* is an exact optimal transport mapping from S to S'.*

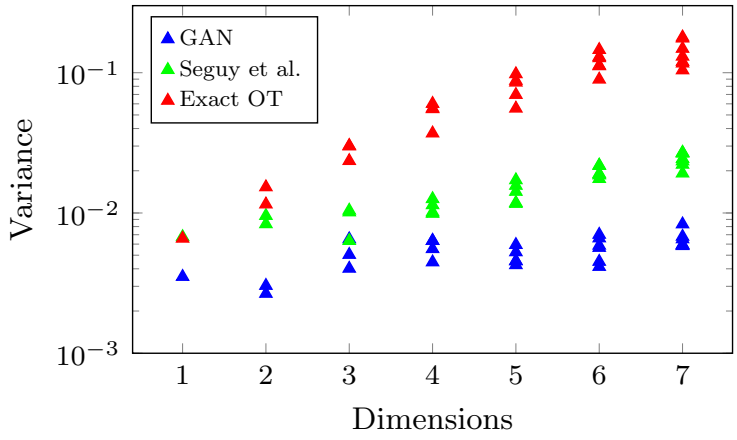


Figure 2.4: Variance of the GAN (blue), Seguy et al. [84] (green), and exact (red) optimal transport mappings over the different random draws of the observed points S and S'. Note the logarithmic vertical scale. The horizontal axis represents the number of dimensions in the feature space $\mathcal{X}$, and each plotted point represents the variance of one of the features.

Proof. If $G(S) = S'$, G induces a bijection between S and S', so we have $\frac{1}{n}\sum_{\mathbf{x}\in S} D(G(\mathbf{x})) = \frac{1}{n}\sum_{\mathbf{x}'\in S'} D(\mathbf{x}')$. Then, (2.4) becomes

$$L_G = \tfrac{1}{n}\textstyle\sum_{\mathbf{x}'\in S'} D(\mathbf{x}') + \tfrac{\lambda}{n}\sum_{\mathbf{x}\in S} c(\mathbf{x}, G(\mathbf{x})).$$

The first term does not depend on G, so if G^* minimizes L_G, it also minimizes $\frac{\lambda}{n}\sum_{\mathbf{x}\in S} c(\mathbf{x}, G(\mathbf{x}))$. Since $\lambda > 0$, G then minimizes $\frac{1}{n}\sum_{\mathbf{x}\in S} c(\mathbf{x}, G(\mathbf{x}))$, which is exactly the definition of an optimal transport mapping. □

Although the generator G will not satisfy $G(S) = S'$ in practice, Theorem 2.8 motivates the use of this generator to approximate an optimal transport mapping. The following experimental results show that the approximate GAN mapping is more stable than the exact mapping, which changes drastically depending on which sets S and S' were drawn from $\mathcal{S}$ and $\mathcal{S}'$.

Stability

Here, we compare the behavior of the exact optimal transport mapping to those of approximate mappings. This is not intended to be a comprehensive evaluation of all approximation methods, but rather an argument for the use of GAN approximations over the exact mapping. Although FlipTest is compatible with any optimal transport method, I argue that GAN approximations tend to give more reliable results.

To measure stability, we fix a point $\mathbf{x} \in S$ and then draw multiple distinct samples of the other $n-1$ points from $\mathcal{S}$ and n points from $\mathcal{S}'$. Thus, we have different sampled sets S and S' each time, and we observe the variance of the point $f(\mathbf{x})$ over the random draws.

The cost function was set to the square of the L^1 distance. The exact optimal transport mapping was computed as a linear program with Gurobi [42] implemented in Python 3, and for the GAN approximation, I trained a Wasserstein GAN [7] using Keras [21] with the TensorFlow backend [1]. For these experiments, I mapped the standard multivariate normal distribution to

itself. Because the size of the linear program for the exact optimal transport mapping increases at least quadratically with the size n of the dataset, I used $n = 500$. Each experiment was repeated with 100 different random draws of the dataset.

In the first set of experiments, I set $\mathbf{x}$ to be the all-ones vector and observed the mean of $f(\mathbf{x})$. Since we map a distribution to itself, f should roughly be the identity function, and the mean of $f(\mathbf{x})$ should be similar to $\mathbf{x}$. While this was the case for the GAN approximation, the exact mapping displayed a significant "regression-to-the-mean" effect that increased with the number of dimensions in the feature space.

In the second set of experiments, I set $\mathbf{x}$ to the zero vector and noted the variance of $f(\mathbf{x})$ under all three types of mappings. The result is plotted in Fig. 2.4, showing that the GAN approximation is much more stable than the exact mapping and somewhat more stable than that obtained by the method of Seguy et al. [84].

The differences in both mean and variance persisted when I changed the data distribution by making the features correlated with each other. These differences can be explained by the fact that the exact mapping tends to overfit to the observed points, since it has to map every point to another observed point. As a result, approximate mappings are better suited for evaluating the fairness of a model that is trained to generalize. Since the GAN mapping appears to be more stable than that of Seguy et al., for the rest of the experiments we will exclusively use GANs as the optimal transport method in FlipTest.

2.3.3 Flipsets and Transparency Reports

We now leverage the optimal transport mapping to gather two main pieces of information from a model: who may experience discrimination, and which features may be associated with this effect. I first describe *flipsets*, which we use to answer the first question, and then show how to use them to construct *transparency reports*, which help answer the second question.

Flipsets

I begin by introducing the *flipset* (Definition 2.5), which is the set of points whose image under a transport mapping is assigned a different label by a binary classifier.

Definition 2.5 (Flipset)**.** *Let $h : \mathcal{X} \to \{0, 1\}$ be a classifier and $G : \mathcal{S} \to \mathcal{S}'$ be an optimal transport mapping (or an approximation thereof). The flipset $F(h, G)$ is the set of points in S whose mapping into $\mathcal{S}'$ under G changes classification.*

$$F(h, G) = \{\mathbf{x} \in S \mid h(\mathbf{x}) \neq h(G(\mathbf{x}))\} \tag{2.5}$$

The positive *and* negative *partitions of $F(h, G)$ are denoted by $F^+(h, G)$ and $F^-(h, G)$.*

$$F^+(h, G) = \{\mathbf{x} \in S \mid h(\mathbf{x}) > h(G(\mathbf{x}))\}$$
$$F^-(h, G) = \{\mathbf{x} \in S \mid h(\mathbf{x}) < h(G(\mathbf{x}))\}$$

In the following experiments, $\mathcal{S}$ and $\mathcal{S}'$ will correspond to two groups with differing values for a protected attribute, and h will be a classifier with the potential to be unfair. For example, suppose that $\mathcal{S}$ and $\mathcal{S}'$ respectively correspond to female and male job applicants and that h is used

to determine which applicants should proceed to further rounds of interview. Then $F^+(h, G)$ is the set of female applicants who proceed to the next round but whose male counterparts under G do not, and $F^-(h, G)$ is the women who do not proceed but whose male counterparts do.

Note that we can also create flipsets based on a mapping $G' : \mathcal{S}' \to \mathcal{S}$ in the opposite direction. Then, in the above example $F^+(h, G')$ is the set of male applicants who proceed to the next round but whose female counterparts under G' do not, and $F^-(h, G')$ is the men who do not proceed but whose female counterparts do. If G' and G compose to the identity function, these flipsets would have the same size as $F^-(h, G)$ and $F^+(h, G)$, respectively, and we can test for this property as a sanity check of our GAN mappings. Table 2.3 describes the results of this sanity check on our experiments in Section 2.3.4.

If the distributions $\mathcal{S}$ and $\mathcal{S}'$ are equal, we would expect G to be the identity function, leading to empty positive and negative flipsets. This corresponds to the setting where the input features are independent of the protected attribute, thereby ensuring that the model cannot discriminate on the basis of the protected attribute. Theorem 2.9, which uses the exact optimal transport mapping to avoid any noise introduced by the GAN approximation, shows that statistical parity only provides a weaker guarantee of zero *net* flipset size. Thus, if the positive and negative flipsets are nonempty but have equal size, some individuals may be experiencing discrimination even though statistical parity holds.

Theorem 2.9. *Let h be a binary classifier and $G : S \to S'$, with $|S| = |S'|$, be the exact optimal transport mapping. Then, $|F^+(h, G)| = |F^-(h, G)|$ if and only if the model satisfies statistical parity on the observed points, i.e.,*

$$|\{\mathbf{x} \in S \mid h(\mathbf{x}) = 1\}| = |\{\mathbf{x}' \in S' \mid h(\mathbf{x}') = 1\}|.$$

Proof. We have

$$\begin{aligned} |F^+(h, G)| &= |\{\mathbf{x} \in S \mid h(\mathbf{x}) = 1 \wedge h(G(\mathbf{x})) = 0\}| \\ &= |\{\mathbf{x} \in S \mid h(\mathbf{x}) = 1\}| - |\{\mathbf{x} \in S \mid h(\mathbf{x}) = 1 \wedge h(G(\mathbf{x})) = 1\}|, \end{aligned}$$

and similarly we have

$$\begin{aligned} |F^-(h, G)| &= |\{\mathbf{x} \in S \mid h(\mathbf{x}) = 0 \wedge h(G(\mathbf{x})) = 1\}| \\ &= |\{\mathbf{x} \in S \mid h(G(\mathbf{x})) = 1\}| - |\{\mathbf{x} \in S \mid h(\mathbf{x}) = 1 \wedge h(G(\mathbf{x})) = 1\}|. \end{aligned}$$

Therefore, the equality $|F^+(h, G)| = |F^-(h, G)|$ is equivalent to $|\{\mathbf{x} \in S \mid h(\mathbf{x}) = 1\}| = |\{\mathbf{x} \in S \mid h(G(\mathbf{x})) = 1\}|$, and it remains to show that the right-hand side of the last equation equals $|\{\mathbf{x}' \in S' \mid h(\mathbf{x}') = 1\}|$. This follows from letting $\mathbf{x}' = G(\mathbf{x})$ and the fact that G is a bijection between S and S'. □

If we instead consider the distributions $\mathcal{S}|(y = 1)$ and $\mathcal{S}'|(y = 1)$, conditioned by the true label y, we can prove a similar result about equality of opportunity [43], and if we consider $\mathcal{S}|(y = 0)$ and $\mathcal{S}'|(y = 0)$ as well, we can extend the result to equalized odds [43].

When h is biased, the flipsets can provide several additional forms useful information about the model's behavior. First, the relative sizes of $F^+(h, G)$ and $F^-(h, G)$ can serve as a simple test of group fairness. Second, the absolute sizes of $F^+(h, G)$ and $F^-(h, G)$, if they are large, can

indicate possible discrimination at the subgroup level. Third, if the distributions of the flipsets are different from $\mathcal{S}$, we gain information about *which* subgroup may be discriminated against. In Section 2.3.4, I apply these insights on case studies with real-world datasets.

Transparency Reports

A *transparency report* (Definition 2.6) identifies features that change the most, and most consistently, under G between members of a given flipset ($F^+(h, G)$ or $F^-(h, G)$). These features are likely candidates for the underlying reasons for the observed discrimination, and can be examined further using more costly causal influence methods [29] to make a final determination.

Definition 2.6 (Transparency Report)**.** *Let $h : \mathcal{X} \to \{0, 1\}$ be a classifier, $G : \mathcal{S} \to \mathcal{S}'$ be an optimal transport mapping (or an approximation thereof), and $F(h, G)$ be the corresponding flipset. If $\mathcal{X} \subseteq \mathbb{R}^d$, we can compute the following vectors, each of whose coordinates corresponds to a feature in $\mathcal{X}$:*

$$\frac{1}{|F^\star(h,G)|} \sum_{\mathbf{x} \in F^\star(h,G)} \mathbf{x} - G(\mathbf{x}), \text{ and}$$
$$\frac{1}{|F^\star(h,G)|} \sum_{\mathbf{x} \in F^\star(h,G)} \operatorname{sign}(\mathbf{x} - G(\mathbf{x}))$$

Here, $\star \in \{+, -\}$. Together, these vectors define a transparency report*, which consists of two rankings of the features in $\mathcal{X}$, each sorted by the absolute value of each coordinate.*

Intuitively, the features ranked highest by the transparency report are those that are most associated with the model's differences in behavior on the flipset. As I show in Section 2.3.4, these often align closely in practice with the features used by the model to discriminate.

2.3.4 Empirical Evaluation of FlipTest

I now apply FlipTest to real and synthetic datasets, illustrating its use in finding discrimination in models. I begin by detailing the experimental setup. After that, I provide additional validation of the GAN optimal transport approximation and then move on to two case studies: a biased predictive policing model, as well as a hiring model that exhibits proxy discrimination. Finally, I compare FlipTest to the related prior work of FairTest by Tramèr et al. [97], arguing that FlipTest can recognize biased features whereas FairTest cannot.

Experimental Setup

The GAN experimental setup is the same for all experiments in this section except λ and the batch size. For all experiments, the cost function was set to the square of the L^1 distance, and a Wasserstein GAN [7] was created using Keras [21] with a TensorFlow backend [1] on Python 3. The Wasserstein GAN has a generator and a critic, each of which has two hidden dense layers of size 128; in addition, the critic's output layer has size 1. Both the discriminator and generator have ReLU activations and were trained with the RMSProp optimizer at a learning rate of 5×10^{-5} for 20,000 epochs. Finally, the critic has a weight clip value of 0.01 and was trained 5 times each time the generator was trained once.

Table 2.2: Validations for GANs included in Section 2.3.4. KS refers to the Kolmogorov–Smirnov two-sample test statistic. MSE Diff refers to the difference, between real and generated data, of the mean squared error of a linear regression model trained on the real data to predict a feature from the rest. For KS and MSE, 10 trials were run; the mean is given first, with standard deviation in parentheses. OT Dist and GAN Dist refer to the average of the squared L^1 distance between a data point x and its counterpart $G(x)$ under an exact optimal transport mapping (OT) and a GAN approximation (GAN), respectively.

Dataset	Features	KS (std)	MSE Diff (std)	OT Dist	GAN Dist
SSL	Age	0.054 (0.001)	0.070 (0.048)	2.04	1.64
	Vic Shooting	0.003 (0.003)	0.108 (0.125)		
	Vic Assault	0.010 (0.002)	0.259 (0.054)		
	Vio Off Arr	0.005 (0.003)	0.065 (0.045)		
	Gang Aff	0.018 (0.006)	0.094 (0.034)		
	Narc Arr	0.025 (0.002)	0.298 (0.058)		
	Trend	0.142 (0.009)	0.368 (0.073)		
	UUW Arr	0.007 (0.002)	−0.103 (0.040)		
Lipton	Work Exp.	0.072 (0.014)	0.150 (0.204)	2.19	2.09
	Hair Len.	0.074 (0.043)	0.141 (0.114)		

For all experiments with the SSL dataset, the GAN was trained on a quarter of the SSL data [24], with eight input features. This corresponds to 41,560 Black subjects and 16,465 White subjects. The random seed was set to 100, the batch size to 4, and $\lambda = 5 \times 10^{-4}$.

For the experiments on the synthetic dataset created by Lipton et al. [65], we have $\lambda = 10^{-4}$ and a batch size of 64. All experiments on this dataset use the random seed 0, and the GAN is trained on 10,000 men and 10,000 women generated from the synthetic data.

GAN Validation

In general, the GAN mapping is an approximation and does not fully converge to the target distribution. Moreover, limitations in the amount of data available to train the GAN will reduce the accuracy of the approximation. Here, I evaluate the effect of these factors on flipsets, concluding that the GAN approximations, although not perfect, are suitable for use with FlipTest.

Control Experiment. First, I trained a GAN with samples from two identical distributions. In this setting, as the sample size approaches infinity, the exact optimal transport mapping will lead to empty flipsets. Therefore, the results of this experiment would indicate how many flips we can expect due to the noise in the GAN approximation and the finite sample size.

Because we map a distribution to itself, in order to simulate a more typical application of GAN training, I added additional random features that are *dependent* on the protected attribute. However, from the perspective of the model, there were no distributional differences between the two protected groups. More specifically, I set $\mathcal{S}$ and $\mathcal{S}'$ respectively to the normal distributions $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{S}}, \mathbf{I}_6)$ and $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{S}'}, \mathbf{I}_6)$, where $\boldsymbol{\mu}_{\mathcal{S}} = (0, 0, 0, 1, 1, 1)^T$, $\boldsymbol{\mu}_{\mathcal{S}'} = (0, 0, 0, -1, -1, -1)^T$, and $\mathbf{I}_6$

is the 6×6 identity matrix. Thus, the first three features had the same joint distribution for both groups, and the model was allowed to use only these three features.

I then constructed the training set by drawing 10,000 points from each of $\mathcal{S}$ and $\mathcal{S}'$. To make the model arbitrary and complicated, I gave each point a binary class label uniformly at random and then trained with this data an SVM classifier with a radial basis function kernel. Finally, I trained a GAN ($\lambda = 10^{-4}$) using a test set, which was drawn from $\mathcal{S}$ and $\mathcal{S}'$ in the same way as the training set, and observed the size of the flipset on the test set.

The GAN correctly mapped the distribution $\mathcal{S}$ to $\mathcal{S}'$, changing each of the first three features by less than 0.01 on average and the other three by close to 2 on average. The model had a very irregular decision boundary, with approximately 63% training accuracy (the test accuracy was, of course, close to 50%) and 53% positive classification rate. However, despite all of the above steps that were intended to increase the size of the flipsets, I observed $|F^+(h, G)| = 167$ and $|F^-(h, G)| = 148$, together accounting for approximately 3% of the data. These numbers stayed similar when I altered $\mathcal{S}$ and $\mathcal{S}'$ so that the features were not necessarily normal and uncorrelated.

This serves a benchmark for comparison with the models in the later experiments, which have larger and more unbalanced flipsets. This difference is striking given that we would expect larger flipsets from the complicated SVM classifier here than the simpler models in the later experiments.

Validations against the Exact Mapping. To further validate the GAN mappings, for each dataset used in the later experiments, I computed the exact optimal transport mapping f on a 2,000-member subset of the data and computed the average squared L^1 distance between x and $f(x)$. Then, I compared this quantity to the average squared L^1 distance between x and $G(x)$ for the GAN generator G. If the GAN mapping closely matches the optimal transport mapping, we would expect these numbers to be similar.

I also examined the fit of the GAN-generated data using the Kolmogorov–Smirnov (KS) two-sample test on the marginal distributions for each feature between the real target data S' and the generated data $G(S)$. The output of this test corresponds to the largest difference in the empirical distribution functions of the two samples, with a small KS statistic corresponding to the case where the two distributions are similar. In Table 2.2, I only report the KS-statistic and its variance, and not the associated p-value, since FlipTest does not require the two samples to be from the exact same distribution. Instead, the KS statistic is simply consulted to make sure that the two samples are not too dissimilar. If the KS statistics are all small, then we can postulate that the GAN has captured the marginals of the target data well.

Since similarity tests on the marginals of a distribution do not account for correlation between features, the output was further validated by training a linear regression model to predict each feature from the others (e.g., in the SSL dataset, age from the other seven features). These regression models were trained on the real target data S', and the accuracies of these models on S' were compared to those on the generated data $G(S)$. If the mean squared error (MSE) is similar between the true data and the generated data, we can take this as evidence that the GAN has captured correlation between the features well. For the SSL and Lipton experiments that follow, the validation results are reported in Table 2.2. The results show that the GANs behave as desired.

Testing a Biased Model (SSL dataset)

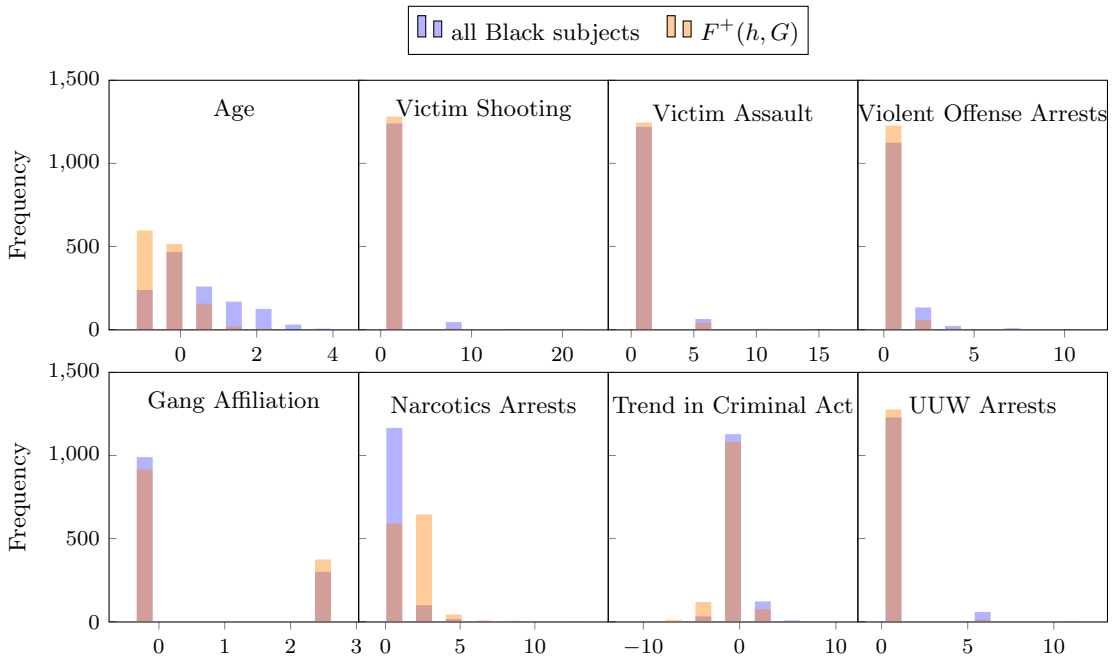
The Chicago Strategic Subject List (SSL) dataset [24] consists of arrest data collected in Chicago for the purpose of identifying which individuals are likely to be involved in a violent crime, either as a victim or a perpetrator. I used the following eight features that are also used by the SSL model: number of times as victim of a shooting incident, age during last arrest, number of times as victim of aggravated battery or assault, number of prior arrests for violent offenses, gang affiliation, number of prior narcotic arrests, trend in recent criminal activity, and number of prior arrests for unlawful use of weapon. The target feature corresponds to the risk of being involved in a shooting, ranging in value from 0 to 500 (low to high risk). All input features were normalized to zero mean and unit variance. A standard least-squares regression model for this data primarily relies on age ($w_{\text{age}} \approx -50$), giving the remaining features coefficients of magnitude less than 10.

A threshold of 345 was set on the risk score to create a classification task; 10% of the dataset has a score above this threshold. As the experiments in Section 2.2.4 show that models trained on this data do not appear to exhibit significant bias, the classifier was deliberately biased to make it rely more heavily on the number of prior narcotics arrests, which is correlated with race ($r = 0.12$) and is arguably less predictive of involvement in violent crime than, say, number of prior arrests for violent offenses. The resulting linear classifier exclusively uses age and narcotics arrests, classifying a person as high risk if and only if $-53 \cdot \text{age} + 25 \cdot \text{narc} > 65$. As a result, the model is expected to assign Black subjects higher average scores without necessarily being accurate.

The positive flipset $F^+(h, G)$ shows that 1,290 Black subjects that are marked by the model as high risk are marked as low risk when sent through the Black-to-White mapping (out of 3,683 Black subjects marked as high risk). On the other hand, the negative flipset $F^-(h, G)$ consists of only 4 people, which is an extremely small fraction of the 37,877 Black subjects marked as low risk. The size of $F^+(h, G)$ and the significant asymmetry of the flipsets suggest that the model discriminates on the basis of race, which is consistent with how the model was constructed. Table 2.3 shows that this asymmetry holds even when the GAN mapping is constructed in the reverse direction from White subjects to Black subjects.

To gain more insight about which subgroups may be discriminated against, we investigate the distributions of the flipsets as compared to that of the general Black population. The histograms of the marginal distributions are given in Fig. 2.5a. For age and narcotic arrests, the flipset subjects skew away from the full population and towards younger people with more narcotic arrests. This suggests the bias of the model affects younger people with more narcotic arrests most. On the other hand, the marginals largely overlap for gang affiliation, which the model does not directly use.

We then look at the transparency report in Fig. 2.5b to learn which features may be responsible for this apparently biased behavior. The Black subjects in $F^+(h, G)$ changed the most, and most consistently, in narcotics arrests under the GAN mapping, which is again consistent with the type of bias that was introduced to this model. Thus, although the transparency reports only provide direct insight into statistical correlations between the features and model behavior, in this case they identify precisely the feature that is responsible for the model's bias. In general, based on the information from the transparency report, an practitioner testing a model can decide whether they would like to investigate the model further for bias based on the features in



(a) Marginal distributions for all Black subjects compared to those of the positive flipset.

Feature	Mean Sign
Narcotic Arr	0.99
Trend	−0.78
Gang Aff	0.26
Vio Off Arr	0.04
Vic Assault	0.03
Age	−0.01
UUW Arr	0.00
Vic Shooting	0.00

Feature	Mean Diff
Narcotic Arr	1.32
Gang Aff	0.71
Trend	−0.20
Vic Assault	0.15
Vio Off Arr	0.11
UUW Arr	0.20
Age	−0.01
Vic Shooting	0.01

(b) Transparency report for the positive flipset. All features are scaled to zero mean and unit variance.

Figure 2.5: Results of the experiment on the biased SSL classifier that puts weight only on age and narcotic arrests.

Table 2.3: Results of flipset sizes based on GAN mappings that are in the reverse direction of those in Section 2.3.4. $|F^+(h, G)|$ and $|F^-(h, G)|$ refer to the sizes of the flipsets that are presented in the text and $|F^-(h, G')|$ and $|F^+(h, G')|$ refer to the sizes of the flipsets based on a mapping in the reverse direction. Also recorded are the values of λ and batch sizes used to create the GAN mapping in the reverse direction; the corresponding values for the forward direction are given at the beginning of Section 2.3.4. As noted in Section 2.3.3, we would expect $|F^+(h, G)| \approx |F^-(h, G')|$ and $|F^-(h, G)| \approx |F^+(h, G')|$, and these equalities approximately hold here.

Dataset	$\lvert F^+(h, G)\rvert$	$\lvert F^-(h, G)\rvert$	$\lvert F^-(h, G')\rvert$	$\lvert F^+(h, G')\rvert$	λ	b
SSL	1290	4	815	3	2×10^{-4}	16
Lipton	715	1215	774	1075	1×10^{-4}	64

the transparency report. A more definitive evidence, such as that given by QII [29], would be required to conclude that the feature causes the model's discriminatory behavior. In Section 2.4 I detail the results of the QII experiments.

Testing a Group-Fair Model (Lipton et al. dataset)

Previously, Lipton et al. [65] argued that some fair learning algorithms employ a "problematic within-class discrimination mechanism". To support this argument, they created a synthetic data distribution that consists of two features: work experience and hair length. They used this distribution to train a learning algorithm by Zafar et al. [110] that seeks to equalize hiring rates across genders. They then found that the resulting linear model uses hair length as a proxy for gender, thereby unfairly benefiting long-haired men and harming short-haired women. Here, I replicate this experimental setting to demonstrate that FlipTest can detect unfair behavior in a model that appears to be fair at the population level.

I trained a linear model on 10,000 men and 10,000 women drawn from the male and female distributions, respectively, after scaling each feature to zero mean and unit variance. Then, I trained approximate optimal transport mappings in both directions and evaluated the fairness of the model on a test set of 10,000 men and 10,000 women drawn from the same distributions. Here we present the women-to-men mapping, and the results on the men-to-women mapping are given in Table 2.3.

At the population level, the model treated the two groups similarly, hiring 27% of the men and 30% of the women. However, when I mapped the women to the men, 715 women were disadvantaged by the model (rejected with hired male counterpart) whereas 1215 were advantaged (hired with rejected male counterpart). These flipsets comprise a much larger portion of the population than those encountered during the GAN validation experiments, suggesting some discriminatory behavior at the subgroup level. Looking more closely at distributions of these flipsets (Figs. 2.3c and 2.3d) provides insight into the subgroups experiencing discrimination: the disadvantaged women tend to have shorter hair and longer work experience than the average woman, and the advantaged women have the opposite characteristics. This is consistent with the observation of Lipton et al. [65] that the model penalizes people with a masculine characteristic (i.e., short hair) in order to equalize hiring rates.

Table 2.4: Results of FairTest [97] on the biased classifier from Section 2.3.4 trained on the SSL dataset. Notably, unlike FlipTest, FairTest does not report much bias based on narcotics arrest.

Subgroup	**Correlation**
All Black population	$[-0.0688, -0.0523]$
Age $\in (-0.94, -0.16]$, Trend $\in (-\infty, 0.50)$	$[-0.1646, -0.1164]$
Age $\in (-\infty, -0.94]$, Trend $\in (-0.98, -0.50)$, Gang Aff. $\in (-\infty, 0.91)$	$[0.0969, 0.1883]$
Vio Off. $\in (-\infty, 0.95]$, Age $\in (-\infty, -0.94]$, Trend $\in (-0.24, \infty)$, Gang Aff. $\in (-\infty, 0.91)$	$[0.0953, 0.1534]$
Vio Off. $\in (-\infty, 0.95]$, Age $\in (-\infty, -0.94]$, Trend $\in (-0.50, \infty)$, Narc Arr. $\in (2.6, \infty)$	$[0.0943, 0.1512]$
Vio Off. $\in (-\infty, 0.95]$, Age $\in (-0.16, \infty]$, Trend $\in (-0.50, \infty)$, Gang Aff. $\in (-\infty, 0.91)$	$[0.0853, 0.3012]$
Age $\in (-\infty, -0.16]$, Trend $\in (-\infty, -0.50)$	$[-0.1203, -0.0804]$
Trend $\in (\infty, -0.50)$	$[-0.1017, -0.0727]$
Vio Off. $\in (-\infty, 0.95]$, Age $\in (-0.95, -0.16]$, Trend $\in (-0.50, \infty)$,	$[-0.0972, -0.0610]$

In addition, the transparency report shows that the disadvantaged women have much less (1.3 standard deviations) work experience and slightly longer (0.5 SD) hair than their counterparts, while the advantaged women have slightly less (0.6 SD) work experience and much longer (1.6 SD) hair than their counterparts. This suggests that the disadvantaged women are disadvantaged due to their short work experience and that the advantaged women are advantaged due to their hair length. In general, this is not sufficient to establish *causal* claims about why the model behaves this way; for example, the difference in hair length may be due to the model's use of some other feature that is correlated with hair length. In this case, however, the model's weights—which define its causal behavior—and the result of the transparency report agree. The model, which is a linear regressor, weights the two features almost equally ($w_{\text{hair}} = 1.4$, $w_{\text{work}} = 1.2$). Since work experience, but not hair length, is a legitimate factor in most hiring decisions, after this deeper investigation into the model we may be able to conclude that this apparently fair model in fact discriminates against some men.

Recognizing Biased Features

Here I demonstrate a major conceptual difference of FlipTest from FairTest by Tramèr et al. [97]. FairTest searches for subgroups experiencing possible discrimination by using a decision tree to iteratively search for subgroups within which there is high correlation between the model output and the protected attribute. It then reports the subgroups with has the highest correlations.

To clearly illustrate the difference between FairTest and FlipTest, we will use a synthetic data distribution with only one feature: the number of prior arrests. This feature is distributed as Geometric$(1/4) - 1$ for people in $\mathcal{S}$ and Geometric$(1/2) - 1$ for people in $\mathcal{S}'$. In addition, we assume a simple model, which classifies a person as low risk if the number of prior arrests is zero, high risk if it is two or more, and either low risk or high risk uniformly at random if there

is exactly one prior arrest. Because the members of $\mathcal{S}$ tend to have more prior arrests than those in $\mathcal{S}'$, this model classifies disproportionately higher number of people in $\mathcal{S}$ as high risk.

I let S and S' be sets of 10,000 points drawn from $\mathcal{S}$ and $\mathcal{S}'$, respectively, and ran both FairTest and FlipTest on these sets. FairTest correctly identified possible discrimination at the group level, reporting a confidence interval of $[0.2452, 0.2941]$ for the correlation between the model's output and the protected attribute. However, the correlation decreased in all subgroups that FairTest subsequently considered. This is because subgroups in FairTest are defined by the values of the input features. Thus, FairTest compares a subgroup of S to a subgroup of S' *that has similar numbers of arrests*.

On the other hand, FlipTest recognizes that the distributions of the number of arrests is different, and adjusts the comparisons accordingly. For example, the set of people with 1 arrest in S is compared to the set with 2–4 arrests in S'. As a result, the flipsets are very large and unbalanced, with $|F^+(h, G)| = 2572$ and $|F^-(h, G)| = 0$. In addition, the transparency report explains a reason for this difference, showing that 100% of the people in $F^+(h, G)$ had more arrests than their statistical counterparts in $\mathcal{S}'$, with an average of 1.44 more arrests.

This phenomenon presents itself on real data as well. On the SSL data, Table 2.4 shows that FairTest identifies an overall bias against the entire Black population. However, it does not report that the discrimination is based on narcotics arrest since the feature itself is biased, with higher levels for Black subjects than White subjects.

Thus, the choice of the appropriate fairness test may depend on the setting. If the number of prior arrests is a strong indicator for recidivism risk, then it makes sense to compare subgroups of people with similar numbers of arrests. On the other hand, if the model had used a feature that is completely unrelated to crime, it would be harder to justify comparing people who are similar with respect to that feature. The experiments here show that FairTest is better suited for settings where the input features can justify potential differences in the model's output.

2.4 Comparison with Black-Box Causal Inference

As mentioned previously, FlipTest does not use any causal information, so it cannot establish that a certain feature or component of a model *causes* the model's discriminatory behavior. In this section, I run experiments computing a causal inference measure called the quantitative input influence (QII) [29], comparing the results of these experiments to FlipTest's transparency reports. Moreover, I propose a modification to QII that reflects the key ideas behind FlipTest, better aligning it to detect the type of discriminatory behavior measured by FlipTest. The comparison between QII and FlipTest shows that while the two measures generally agree, transparency reports are not as sensitive to the exact features that the model uses.

Because the runtime of QII is exponential in the number of input features, when there is a moderate-to-large number of features, it may not be feasible to analyze every feature with QII. On the other hand, FlipTest is unable to ascertain any causal information that may be critical for determining how to mitigate the discriminatory behavior. The results in this section suggest a best-of-both-worlds alternative: We can first apply FlipTest and use its transparency reports to identify features of interest, and then apply QII on this subset of features in order to establish causation.

2.4.1 Local Quantitative Input Influence

I now present a brief summary of the local QII measure, referring the reader to the original paper by Datta et al. [29] for a more detailed exposition. The local QII measure quantifies the contribution of each input feature of a model to the model's output on a given point $\mathbf{x}$. Formally, let h be a classification model, and let $\mathcal{D}$ be a distribution of the input features. Then, for each subset T of the input feature set, we compute the following value, which we can interpret as the influence of the subset:

$$\iota(T) = \Pr_{\mathbf{u} \sim \mathcal{D}}[h(\mathbf{x}) \neq h(\mathbf{x}_{-T}\mathbf{u}_T)] \tag{2.6}$$

Here, $\mathbf{x}_{-T}\mathbf{u}_T$ is a new point created through a causal intervention where the values of the features in the set T are taken from the point $\mathbf{u}$ and the rest are taken from $\mathbf{x}$. Thus, $\mathbf{x}_{-T}\mathbf{u}_T$ satisfies

$$(\mathbf{x}_{-T}\mathbf{u}_T)_i = \begin{cases} \mathbf{u}_i & \text{if } i \in T \\ \mathbf{x}_i & \text{otherwise.} \end{cases}$$

Afterwards, the Shapley value [86] is used to compute the contribution of each input feature separately.

Recall that FlipTest's transparency reports identify the features that change the most between a specific subgroup (flipset) and its counterpart. Intuitively, the transparency reports suggest features that could be responsible for the potentially discriminatory behavior of the model. By applying local QII to the members of the flipset, we can evaluate whether these features are *causally* responsible.

It now remains to specify the distribution $\mathcal{D}$. In the original work, Datta et al. drew the causal intervention point $\mathbf{u}$ from the entire population distribution, and I follow this convention in one set of my experiments. However, under this convention it is possible for $\mathbf{x}$ and $\mathbf{u}$ to belong to the same protected group, whereas FlipTest always maps $\mathbf{x} \in \mathcal{S}$ to a point $G(\mathbf{x}) \in \mathcal{S}'$ that belongs to a different protected group. Therefore, for better comparison with FlipTest, I also run another set of experiments where $\mathbf{x} \in \mathcal{S}$ is causally intervened on with point $\mathbf{u}$ drawn from the distribution $\mathcal{S}'$.

2.4.2 Linear Models

We first analyze the Chicago SSL dataset [24] and the synthetic dataset created by Lipton et al. [65], to which we applied FlipTest in Section 2.3.4. For Chicago SSL, Fig. 2.5a shows that the Black subjects in the positive flipset tends to have lower age and higher narcotics arrests than the general Black population. Moreover, the transparency report in Fig. 2.5b suggests that these people may be discriminated against because of narcotics arrests, as well as gang affiliation and negative trend in criminal act. For the synthetic dataset, Fig. 2.3b and the corresponding transparency report in Section 2.3.4 suggest that the women in the positive flipset are advantaged due to their long hair, whereas the women in the negative flipset are disadvantaged due to their relative lack of work experience. I now apply local QII [29] to evaluate whether these transparency reports comport with the causal behavior of the models.

For the Chicago SSL dataset, I filtered by age and narcotics arrests for the characteristics of the positive flipset. In particular, following the marginal distributions in Fig. 2.5a, I looked

SSL $F^+(h, G)$

Feature	Infl.
Narcotic Arr	0.521
Age	0.426
Vic Shooting	0.000
Vic Assault	0.000
Vio Off Arr	0.000
Gang Aff	0.000
Trend	0.000
UUW Arr	0.000

Lipton $F^+(h, G)$

Feature	Infl.
Hair Len	0.694
Work Exp	0.031

Lipton $F^-(h, G)$

Feature	Infl.
Hair Len	0.227
Work Exp	0.049

Adult $F^+(h, G)$

Feature	Infl.
Education	0.414
Marital Stat	0.315
Cap Gain	0.059
Hrs per Wk	0.057
Cap Loss	0.013
Age	0.002
Workclass	0.000
Occupation	0.000
Relationship	0.000
Native Ctry	0.000

(a) QII feature influences when the causal intervention point ($\mathbf{u}$ in Eq. (2.6)) is drawn from the entire population distribution.

SSL $F^+(h, G)$

Feature	Infl.
Narcotic Arr	0.549
Age	0.408
Vic Shooting	0.000
Vic Assault	0.000
Vio Off Arr	0.000
Gang Aff	0.000
Trend	0.000
UUW Arr	0.000

Lipton $F^+(h, G)$

Feature	Infl.
Hair Len	0.835
Work Exp	−0.084

Lipton $F^-(h, G)$

Feature	Infl.
Work Exp	0.201
Hair Len	0.048

Adult $F^+(h, G)$

Feature	Infl.
Marital Stat	0.433
Education	0.334
Hrs per Wk	0.077
Cap Gain	0.075
Cap Loss	0.015
Age	0.002
Workclass	0.000
Occupation	0.000
Relationship	0.000
Native Ctry	0.000

(b) QII feature influences when the causal intervention point ($\mathbf{u}$ in Eq. (2.6)) is drawn only from the protected group corresponding to $\mathcal{S}'$.

Figure 2.6: Feature influences computed using local QII on subgroups identified through FlipTest. The subgroups were chosen based on the marginal distributions of the flipsets.

at the Black people who are classified as high risk and satisfy $\mathrm{age} < -1$ and $\mathrm{narc} > 0$. For the synthetic dataset, I referred to the marginal distributions in Figs. 2.3c and 2.3d to identify advantaged and disadvantaged subgroups whose characteristics match those of the positive and negative flipsets, respectively. The advantaged subgroup consists of hired women with $-1 < \mathrm{work\_exp} < 0$ and $\mathrm{hair\_len} > 1$, and the disadvantaged subgroup consists of rejected women such that $0 < \mathrm{work\_exp} < 1$ and $\mathrm{hair\_len} < 0$. (Recall that all input features were normalized to zero mean and unit variance.) Then, I computed the mean feature influences for members of these subgroups using local QII, taking a random sample of 1,000 people if the subgroup was too large.

The results of the experiments are given in Fig. 2.6. For the Chicago SSL dataset, the only features that have nonzero influence are narcotics arrests and age, which are the only two features that our model uses. Thus, whereas the transparency report in Fig. 2.5b may have led one to believe that the model uses the criminal trend and gang affiliation features to discriminate, QII shows that the model does not *directly* use them. This does not necessarily mean that there is no issue in the model with respect to gang affiliation, but it sheds light on the *cause* of the model's behavior so that concerned parties can more effectively address any issues that may exist.

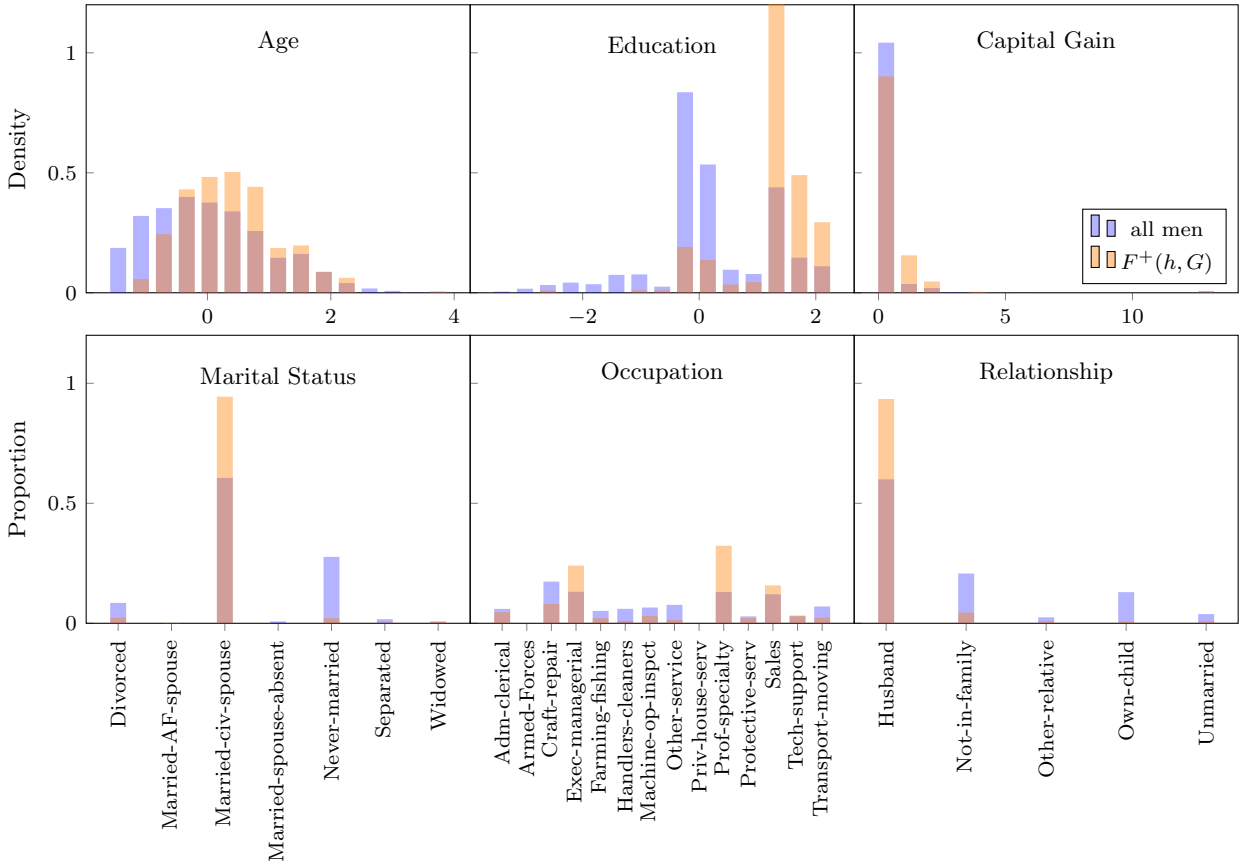
For the synthetic dataset, a significant difference results from the choice of the distribution $\mathcal{D}$ in Eq. (2.6). If the causal intervention point $\mathbf{u}$ is drawn from the entire population distribution, QII reports that hair length is several times more influential than work experience in the outcome of the disadvantaged women. This is because a woman in the negative flipset tends to have above-average work experience, so replacing her work experience with that of a random person is not likely to flip her outcome from rejected to hired. On the other hand, in this dataset the average man has significantly more work experience than the average person, so if $\mathcal{D}$ is the male distribution, QII assigns a much higher influence to work experience than hair length. The transparency report is consistent with this latter result, as women in the negative flipset tend to have male counterparts with much more work experience.

Another noteworthy result is that the mean influence of work experience in Fig. 2.6b is negative. This arises because the effect of intervening on hair length is extremely strong. Thus, intervening on both features together is less likely to flip the output of the model than intervening on hair length alone, and the Shapley value assigns a negative contribution to work experience.

2.4.3 Non-Linear Model

The results in the previous section show that the QII feature influences mostly agree with the transparency reports for the SSL and Lipton models. However, both of these models are linear, so there are limited ways in which an input to one of these models can affect its output. For more complex models, it is possible that FlipTest and QII behave more differently, so I now evaluate a non-linear model.

In particular, I compare FlipTest and QII on a decision tree trained with the UCI Adult [33] dataset, which comes from the 1994 Census database and consists of 14 features. Of these features, I removed race and sex to avoid directly using protected demographic attributes, the categorical education feature because it was redundant with a numerical feature containing the same information, and final weight. I then trained a decision tree with the remaining ten features, five numerical (age, education, capital gain, capital loss, and hours per week) and five categorical



(a) Marginal distributions for all men compared to those of the positive flipset. There were no significant differences in the marginal distributions of the features that are not present in this figure.

Feature	Mean Sign
Age	−0.51
Hrs per Wk	−0.47
Education	−0.25
Cap Gain	−0.23
Cap Loss	−0.08

Feature	Mean Diff
Hrs per Wk	−0.36
Cap Gain	−0.27
Age	−0.24
Cap Loss	−0.18
Education	−0.16

(b) Transparency report for the positive flipset. Only the numerical features are given here, as the transparency report is not defined for categorical features. All numerical features are scaled to zero mean and unit variance.

Figure 2.7: Results of the FlipTest experiment on the UCI Adult decision tree.

(class of work, marital status, occupation, relationship in household, and native country), to predict whether a person makes over $50,000 per year.

For FlipTest, because GANs do not perform as well on categorical features [17], I instead computed the exact optimal transport mapping from men to women on a test set of 3,000 men and 3,000 women. The predictions of the model on this test set are biased with respect to gender, as the model classifies 695 men but only 182 women as high income. The asymmetry in the flipset sizes ($|F^+(h, G)| = 520$ and $|F^-(h, G)| = 7$) were consistent with this bias. Fig. 2.7 shows the marginal distributions and transparency reports for the positive flipset $F^+(h, G)$. The two forms of transparency reports, mean change in sign and mean difference, do not agree on the relative importance of the features, so the local QII measure would be especially helpful in explaining the behavior of this model.

For QII, to obtain a subgroup whose characteristics match those of the positive flipset, I filtered for the men who are predicted to have high income and satisfy $\mathrm{education} > 1$ and $\mathrm{marital\_status} =$ "Married-civ-spouse". The experimental results were similar when I filtered for $\mathrm{occupation} \in \{$"Exec-managerial", "Prof-specialty"$\}$ or $\mathrm{relationship} =$ "Husband" in lieu of $\mathrm{marital\_status}$.

The two feature rankings in Fig. 2.6 agree that education and marital status are the most influential, but the relative importance of these two features depends on whether the causal intervention point $\mathbf{u}$ is drawn from the entire population distribution or just the female distribution. This is because the root node of the decision tree branches on whether $\mathrm{marital\_status} =$ "Married-civ-spouse". In this dataset, a much higher fraction of the men fall under this category than the women, so when $\mathbf{u}$ corresponds to a woman, the causal intervention on marital status is significantly more likely to change the branch taken, increasing the likelihood to flip the output of the decision tree.

Additionally, the QII-based feature rankings contrast with the transparency report-based rankings in Fig. 2.7b, where education is the third and fifth most important feature, respectively, out of the five numerical features. This suggests that the causal behavior of a model can differ considerably from what can be inferred through statistical methods alone. For example, because marital status and relationship are mostly redundant features in this dataset, if the root node had branched on $\mathrm{relationship} =$ "Husband" instead of $\mathrm{marital\_status} =$ "Married-civ-spouse", the outputs of the new tree, and by extension the transparency report, would have been very similar. On the other hand, the QII measures on this new tree would have been very different, assigning high influence to relationship and none to marital status. Thus, I recommend that practitioners use statistical methods, such as FlipTest, and causal methods, such as QII, together in order to gain a holistic understanding of a model.

Chapter 3

Individual Fairness Revisited

Another way in which a model can be unfair is epitomized by the example of arbitrary predictions in Chapter 1. Namely, a model can trivially satisfy statistical parity by arbitrarily assigning outcomes to people. This is because statistical parity only prohibits a disparity between two demographic groups, forgoing any restrictions on how it treats two people from the same group. Instead, the prohibition on arbitrary behavior is better captured by *individual fairness* [34], which imposes a continuity requirement on models to ensure that similar people are treated similarly. In this chapter, I present black-box methods for individual fairness, extending a meaningful individual fairness guarantee to every model, even ones for which continuity analysis is intractable.

Like in the previous chapter, I begin by considering linear models, which have no meaningful distinction between black-box and white-box access due to Gaussian elimination. One challenge with operationalizing individual fairness is that it raises the difficult societal question of how to define which people are "similar". In Section 3.2, I address this issue with the insight that it may be easier to determine whether a given similarity metric is reasonable than it is to construct one from scratch. Thus, rather than imposing individual fairness with a predetermined similarity metric, we find a metric that corresponds to the behavior of a given model, which can then guide the discussion on whether the model is fair. To facilitate this, I introduce the notion of a *minimal* fairness metric, showing that for any linear and logistic regression model there exists a unique metric that best characterizes the behavior of the model.

In Section 3.3, I consider more complicated models, such as deep neural networks, whose minimal metrics are not easily computable. I show that, with only black-box access to any given model, we can make the model provably individually fair by post-processing it with the technique of *randomized smoothing* by Cohen et al. [25]. Compared to the results of Cohen et al., the theorems in this chapter are in a sense stronger because individual fairness is a uniform requirement that applies to all points in the input space, whereas the certified threshold of Cohen et al. is a function of the input point. I present Laplace and Gaussian smoothing mechanisms, which are versatile in that they can make a model provably individually fair under any given weighted L^p metric. Moreover, I argue that these smoothing mechanisms do not add more noise than is necessary by proving the minimality of this metric for the smoothed model.

Finally, the experiments in Section 3.5 combine the two main ideas in this chapter—I smooth neural networks to be individually fair under the minimal metrics of linear models trained on the same datasets. The results on four real datasets show that the neural networks smoothed

with Gaussian noise in particular are often approximately as accurate as the original models. Moreover, the smoothed neural networks have the same favorable individual fairness guarantees as those of linear models and sometimes benefit from the increased predictive accuracy enabled by the neural network. However, the experiments also show that there is a trade-off between fairness and utility, so in some settings it may be desirable to aim for an even higher accuracy by relaxing the fairness guarantees.

The contents of this chapter are largely based on a previously published work at the International Joint Conference on Artificial Intelligence [105].

3.1 Related Work

Dwork et al. [34] introduced the definition of individual fairness, which contrasts with group-based notions of fairness [43, 109] that require demographic groups to be treated similarly on average. Motivated in part by group fairness, Zemel et al. [111] learn a representation of the data that excludes information about a protected attribute, such as race or gender, whose use is often legally prohibited. This work has spurred more research on fair representations [16, 70, 95], and the resulting representations implicitly define a similarity metric. However, unlike the weighted L^p metrics that I use, these metrics are harder for humans to interpret and are primarily designed to attain group fairness.

Others approximate individual fairness based on a limited number of oracle queries, which represent human judgments, about whether pair of individuals is similar. Gillen et al. [38] attempt to learn a similarity metric that is consistent with the human judgments in the setting of online linear contextual bandits. In a more general setting, Ilvento [48] derives an approximate metric using comparison queries that ask which of two individuals a given third individual is more similar to. Finally, Jung et al. [53] apply constrained optimization directly without assuming that the human judgments are consistent with a metric.

By contrast, I post-process a model using randomized smoothing to provably ensure individual fairness. Cohen et al. [25] previously analyzed randomized smoothing in the context of adversarial robustness. In the context of fairness, most post-processing approaches [18, 43] do not take individual fairness into account, and although Lohia et al. [68] consider individual fairness, they define two individuals to be similar if and only if they differ only in the pre-specified protected attribute.

3.2 Theoretical Analysis of Minimal Metrics

I begin with some preliminary definitions and notation that we will use throughout this chapter.

Definition 3.1 (Distance metric)**.** *A nonnegative function $D : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a* distance metric *in $\mathcal{X}$ if it satisfies the following three conditions: nonnegativity, symmetry, and triangle inequality.*

In common mathematical usage, Definition 3.1 is a pseudometric, and metrics must also satisfy the condition that $D(x_1, x_2) = 0$ if and only if $x_1 = x_2$. However, I will refer to pseudometrics as metrics, following the convention in the field of metric learning.

For the definition of individual fairness (Definition 3.2), some more notation is necessary. Let $\mathcal{X}$ and $\mathcal{Y}$ denote a model's input and output spaces, respectively. Moreover, we will assume a distance metric $D_{\mathcal{Y}} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ that characterizes how close two points in the output space are.

Definition 3.2 (Individual fairness [34])**.** *A model $h : \mathcal{X} \to \mathcal{Y}$ is* individually fair *under metric $D_{\mathcal{X}} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ if, for all $x_1, x_2 \in \mathcal{X}$,*

$$D_{\mathcal{Y}}(h(x_1), h(x_2)) \leq D_{\mathcal{X}}(x_1, x_2). \tag{3.1}$$

Individual fairness formalizes the intuition that the model should not behave arbitrarily. In particular, it requires that similar individuals should be treated similarly, i.e., given two individuals $x_1, x_2 \in \mathcal{X}$, if the distance $D_{\mathcal{X}}(x_1, x_2)$ between them is small, then the distance $D_{\mathcal{Y}}(h(x_1), h(x_2))$ between the outputs of the model on these individuals should also be small.

3.2.1 Definition of a Minimal Metric

One criticism of individual fairness is that it is difficult to apply in practice because it requires one to specify the metric $D_{\mathcal{X}}$ [23]. The choice of a metric in $\mathcal{X}$ dictates which individuals should be considered similar, which is highly context-dependent and often controversial. Thus, I take a slightly different approach—rather than specifying a metric $D_{\mathcal{X}}$ and asking whether a model is individually fair under that metric, we find one metric under which the model is individually fair. Then, we can reason about whether the metric is appropriate for the task at hand.

However, there could be multiple metrics for which a model is individually fair. In fact, if $D_{\mathcal{X}}(x_1, x_2) \geq D'_{\mathcal{X}}(x_1, x_2)$ for all $x_1, x_2 \in \mathcal{X}$, then any model that is individually fair under $D'_{\mathcal{X}}$ is also fair under $D_{\mathcal{X}}$, as the metrics are simply upper bounds on the extent to which a model's outputs can vary. On the other hand, our goal is to describe the behavior of a model, for which we need a *tight* upper bound. This notion of tightness is captured by the minimality of a distance metric, defined in Definition 3.3.

Definition 3.3 (Minimal distance metric)**.** *Let $\mathcal{M}$ be a set of distance metrics in $\mathcal{X}$. A metric $D_{\mathcal{X}} \in \mathcal{M}$ is* minimal *in $\mathcal{M}$ with respect to model $h : \mathcal{X} \to \mathcal{Y}$ if* (1) *h is individually fair under $D_{\mathcal{X}}$, and* (2) *there does not exist a different $D'_{\mathcal{X}} \in \mathcal{M}$ such that h is individually fair under $D'_{\mathcal{X}}$ and $D_{\mathcal{X}}(x_1, x_2) \geq D'_{\mathcal{X}}(x_1, x_2)$ for all $x_1, x_2 \in \mathcal{X}$.*

3.2.2 Theorems about Unique Minimal Metrics

I now present Theorems 3.1, 3.2, and 3.4, which state that in some cases there exists a unique minimal metric with respect to a model. Then, the unique minimal metric is in a sense the metric that best characterizes the behavior of the model, so we can reason about the metric to help decide whether the model is fair.

Theorem 3.1. *Let $h : \mathcal{X} \to \mathcal{Y}$ be any model, and let $\mathcal{M}_{\text{all}}$ be the set of all metrics that satisfy the conditions in Definition 3.1. Then, the metric $D_{\mathcal{X}}$, defined as*

$$D_{\mathcal{X}}(x_1, x_2) = D_{\mathcal{Y}}(h(x_1), h(x_2)) \tag{3.2}$$

for all $x_1, x_2 \in \mathcal{X}$, is the unique minimal metric in $\mathcal{M}_{\text{all}}$ with respect to h.

Proof. I first prove that $D_{\mathcal{X}}$ is a minimal metric, and later I will prove that no other metric is minimal. Since we assume $D_{\mathcal{Y}}$ to be a metric, it easily follows that $D_{\mathcal{X}}$ is also a metric under Definition 3.1. Moreover, the equality in Eq. (3.1) always holds by the definition of $D_{\mathcal{X}}$, so h is individually fair under $D_{\mathcal{X}}$. Thus, it remains to show that there does not exist a different $D'_{\mathcal{X}} \in \mathcal{M}_{\text{all}}$ such that h is individually fair under $D'_{\mathcal{X}}$ and $D_{\mathcal{X}}(x_1, x_2) \geq D'_{\mathcal{X}}(x_1, x_2)$ for all $x_1, x_2 \in \mathcal{X}$.

Suppose such $D'_{\mathcal{X}}$ exists. Since $D'_{\mathcal{X}} \neq D_{\mathcal{X}}$, there must exist some $x_1, x_2 \in \mathcal{X}$ such that $D_{\mathcal{X}}(x_1, x_2) > D'_{\mathcal{X}}(x_1, x_2)$. This, combined with Eq. (3.2), contradicts the assumption that h is individually fair under $D'_{\mathcal{X}}$.

Now I prove that $D_{\mathcal{X}}$ is the unique minimal metric, arguing that $D'_{\mathcal{X}}$ cannot be minimal if $D'_{\mathcal{X}} \neq D_{\mathcal{X}}$. If there exist $x_1, x_2 \in \mathcal{X}$ such that $D_{\mathcal{X}}(x_1, x_2) > D'_{\mathcal{X}}(x_1, x_2)$, then h is not individually fair under $D'_{\mathcal{X}}$. Thus, we must have $D'_{\mathcal{X}}(x_1, x_2) \geq D_{\mathcal{X}}(x_1, x_2)$ for all $x_1, x_2 \in \mathcal{X}$, but then h is individually fair under $D_{\mathcal{X}}$, so $D'_{\mathcal{X}}$ cannot be minimal. □

Theorem 3.1 shows that the minimal metric $D_{\mathcal{X}}$ in $\mathcal{M}_{\text{all}}$ is defined directly in terms of the model in question. However, the minimal metric should ideally be simpler than the model so that it can help us interpret and reason about the fairness of the model. Thus, from now on we will only consider weighted L^p metrics (Definition 3.4), which comprise a broad and interpretable family of metrics defined over $\mathbb{R}^d$.

Definition 3.4 (Weighted L^p metric)**.** *The* weighted L^p metric*, with $p \geq 1$ and weights $w_i \geq 0$, is a distance metric in $\mathbb{R}^d$ that is defined by the equation*

$$D(\mathbf{x}_1, \mathbf{x}_2) = \left(\sum_{i=1}^{d} w_i \cdot |x_{1i} - x_{2i}|^p \right)^{1/p}, \tag{3.3}$$

where x_{1i} and x_{2i} are the i-th coordinates of $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively.

The restriction that $p \geq 1$ exists because otherwise the function D does not satisfy the triangle inequality.

With this set of metrics, we can no longer prove a theorem as general as Theorem 3.1. Therefore, I instead prove a result for linear regression models. In the linear regression setting, we have $\mathcal{Y} = \mathbb{R}$, and the distance metric is simply the absolute value $D_{\mathcal{Y}}(y_1, y_2) = |y_1 - y_2|$. Theorem 3.2 identifies the weighted L^p metric that is uniquely minimal for a given linear regression model.

Theorem 3.2. *Let $h : \mathbb{R}^d \to \mathbb{R}$ be a linear regression model with coefficients $\beta_1, \ldots, \beta_d$, and let $\mathcal{M}_L$ be the set of all weighted L^p metrics. Then, the L^1 metric $D_{\mathcal{X}}$, with weights $w_i = |\beta_i|$, is the unique minimal metric in $\mathcal{M}_L$ with respect to h.*

Proof. $D_{\mathcal{X}}$ is clearly in $\mathcal{M}_L$ by definition. To see that h is individually fair under $D_{\mathcal{X}}$, note that for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$

$$D_{\mathcal{Y}}(h(\mathbf{x}_1), h(\mathbf{x}_2)) = \left| \sum_{i=1}^{d} \beta_i (x_{1i} - x_{2i}) \right| \leq \sum_{i=1}^{d} |\beta_i (x_{1i} - x_{2i})| = D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2). \tag{3.4}$$

The rest of the proof closely mirrors the argument given in the proof of Theorem 3.1, so I only mention how the proofs differ. As in the proof of Theorem 3.1, we assume that there

exist $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$ such that $D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) > D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2)$. The goal is to show that $D'_{\mathcal{X}}$ is not individually fair, and for this proof we have the additional condition that $D'_{\mathcal{X}} \in \mathcal{M}_L$. However, it is not necessarily true that $D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) < D_{\mathcal{Y}}(h(\mathbf{x}_1), h(\mathbf{x}_2))$, so we instead construct $\mathbf{x}'_2$ such that $D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}'_2) < D_{\mathcal{Y}}(h(\mathbf{x}_1), h(\mathbf{x}'_2))$.

Let $x'_{2i} = x_{1i} - \operatorname{sgn}(\beta_i)|x_{1i} - x_{2i}|$. With Eq. (3.3), we can verify that $D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) = D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}'_2)$ for any $D_{\mathcal{X}} \in \mathcal{M}_L$. Moreover, $\beta_i(x_{1i} - x'_{2i}) \geq 0$ for all i, so the equality in Eq. (3.4) holds if we replace $\mathbf{x}_2$ by $\mathbf{x}'_2$. Combining all of these relations, we arrive at the desired result:

$$D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}'_2) = D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) < D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) = D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}'_2) = D_{\mathcal{Y}}(h(\mathbf{x}_1), h(\mathbf{x}'_2)). \qquad \square$$

The final results in this section deals with logistic regression models. I first apply Lemma 3.3 to derive the minimal metric of a logistic regression model. Then, I prove Theorem 3.4, which shows that the derived metric is in fact the unique minimal metric.

Lemma 3.3. *Let $h_1 : \mathcal{X} \to \mathbb{R}$ and $h_2 : \mathbb{R} \to \mathbb{R}$ be functions, and suppose $|h'_2(x)| \leq c$ for all x. If h_1 is individually fair under $D_{\mathcal{X}}$, then $h_2 \circ h_1$ is individually fair under $c \cdot D_{\mathcal{X}}$.*

Proof. By the individual fairness of h_1, we have

$$|h_1(\mathbf{x}_1) - h_1(\mathbf{x}_2)| = D_{\mathcal{Y}}(h_1(\mathbf{x}_1), h_1(\mathbf{x}_2)) \leq D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2)$$

for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$. Moreover, because h_2 has a first derivative that is bounded by c, it is c-Lipschitz continuous. Thus, for all $\mathbf{x}_1$ and $\mathbf{x}_2$ we have

$$|h_2(h_1(\mathbf{x}_1)) - h_2(h_1(\mathbf{x}_2))| \leq c \cdot |h_1(\mathbf{x}_1) - h_2(\mathbf{x}_2)|,$$

and the result follows from combining the above two inequalities. $\square$

The confidence of a logistic regression model can be expressed as $h_2(h_1(\mathbf{x}))$, where h_1 is a linear regression model and $h_2(x) = e^x/(1 + e^x)$. By Theorem 3.2, h_1 is individually fair under the L^1 metric whose weights are the regression coefficients. Since $|h'_2(x)|$ is bounded by $\frac{1}{4}$, Lemma 3.3 implies that the logistic regression models are individually fair under a metric whose weights are a quarter times the regression coefficients. Theorem 3.4 shows that this is in fact the unique minimal metric.

Theorem 3.4. *Let $h : \mathbb{R}^d \to [0, 1]$ be a logistic regression model with coefficients $\beta_1, \ldots, \beta_d$, and let $\mathcal{M}_L$ be the set of all weighted L^p metrics. Then, the L^1 metric $D_{\mathcal{X}}$, with weights $w_i = \frac{1}{4}|\beta_i|$, is the unique minimal metric in $\mathcal{M}_L$ with respect to h.*

Proof. I have already shown that h is individually fair under $D_{\mathcal{X}}$, so it remains to be shown that $D_{\mathcal{X}}$ is uniquely minimal. As before, we assume that there exist $D'_{\mathcal{X}} \in \mathcal{M}_L$ and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$ such that $D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) > D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2)$, and we aim to show that h is not individually fair under $D'_{\mathcal{X}}$.

Let us decompose h as $h_2 \circ h_1$, where h_1 is the linear regression model with the weights β_i, and $h_2(x) = e^x/(1 + e^x)$. Following the argument in the proof of Theorem 3.2, we can find $\mathbf{x}'_2$ such that

$$D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}'_2) < D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}'_2) = \frac{1}{4}|h_1(\mathbf{x}_1) - h_1(\mathbf{x}'_2)|.$$

To show that h is not individually fair, we will use $\mathbf{x}_1$ and $\mathbf{x}'_2$ to construct $\mathbf{x}''_1$ and $\mathbf{x}''_2$ such that

$$D'_{\mathcal{X}}(\mathbf{x}''_1, \mathbf{x}''_2) < D_{\mathcal{X}}(\mathbf{x}''_1, \mathbf{x}''_2) = \tfrac{1}{4}|h_1(\mathbf{x}''_1) - h_1(\mathbf{x}''_2)| \approx |h(\mathbf{x}''_1) - h(\mathbf{x}''_2)|. \tag{3.5}$$

First, find $\mathbf{x}''_1$ such that $h_1(\mathbf{x}''_1) = 0$. This is always possible unless the regression coefficients β_i are all zero, in which case the theorem statement is trivially true. In addition, let $\mathbf{x}''_2 = \mathbf{x}''_1 + \epsilon(\mathbf{x}'_2 - \mathbf{x}_1)$, where ϵ is a small constant. Then, by the translation invariance and scalar multiplication properties of L^p metrics, we have

$$D_{\mathcal{X}}(\mathbf{x}''_1, \mathbf{x}''_2) = D_{\mathcal{X}}(0, \epsilon(\mathbf{x}'_2 - \mathbf{x}_1)) = \epsilon \cdot D_{\mathcal{X}}(0, \mathbf{x}'_2 - \mathbf{x}_1) = \epsilon \cdot D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}'_2),$$

and similarly $D'_{\mathcal{X}}(\mathbf{x}''_1, \mathbf{x}''_2) = \epsilon \cdot D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}'_2)$. By the same two properties, we also have $|h_1(\mathbf{x}''_1) - h_1(\mathbf{x}''_2)| = \epsilon \cdot |h_1(\mathbf{x}_1) - h_1(\mathbf{x}'_2)|$, so the first two (in)equalities of Eq. (3.5) hold.

Finally, note that $h'_2(0) = 1/4$. Because $h_1(\mathbf{x}''_1) = 0$, this means that $|h(\mathbf{x}''_1) - h(\mathbf{x}''_2)| = |h_2(h_1(\mathbf{x}''_1)) - h_2(h_1(\mathbf{x}''_2))|$ approaches $\frac{1}{4}|h_1(\mathbf{x}''_1) - h_1(\mathbf{x}''_2)|$ as $\epsilon \to 0$. Thus, if ϵ is sufficiently small, h is not individually fair under $D'_{\mathcal{X}}$, which is what we wanted. □

3.3 Randomized Smoothing

For more complex models, the minimal metrics are more difficult to compute. Moreover, even when the minimal metric of a model can be computed, the model may not be individually fair under any reasonably sized metric. For example, neural networks are often susceptible to adversarial examples [40, 92], which are inputs to the model that are created by applying a small perturbation to an original input with the goal of causing a very large change in the model's output. The frequent success of these attacks show that a small change in $\mathcal{X}$ can cause a large change in $\mathcal{Y}$, which is contrary to individual fairness.

Therefore, for general models I take a different approach. Previously, Cohen et al. [25] introduced randomized smoothing, a post-processing method that ensures that the post-processed model is robust against perturbations of size, measured with the standard L^2 norm, up to a threshold that depends on the input point. In this section, I prove that I can apply a modified version of randomized smoothing to make the model individually fair. This result does not immediately follow from prior results—individual fairness imposes the same constraint on every point in the input space, whereas the certified threshold of Cohen et al. is a function of the input point.

3.3.1 Definition of Randomized Smoothing

In this section, we assume that $\mathcal{Y}$ is categorical, following the setting of Cohen et al. [25]. I present two post-processing methods for deriving an individually fair model from an arbitrary function $f : \mathbb{R}^d \to \mathcal{Y}$. Like the models considered by Dwork et al. [34], the post-processed fair model $h_{f,g}$ maps $\mathbb{R}^d$ to $\Delta(\mathcal{Y})$, which is the set of probability distributions over $\mathcal{Y}$. It is important to note that $h_{f,g}$ is deterministic and that we treat its output simply as an array of probabilities. Thus, to avoid confusion with the randomness that I introduce in Section 3.4, I will write $h_{f,g}(\mathbf{x})[y]$ to denote the probability $\Pr[h_{f,g}(\mathbf{x}) = y]$.

Definition 3.5 (Randomized smoothing)**.** *Let $f : \mathbb{R}^d \to \mathcal{Y}$ be an arbitrary model, and let $g : \mathbb{R}^d \to \mathbb{R}$ be a probability distribution.*[4] *Then, the* smoothed model $h_{f,g} : \mathbb{R}^d \to \Delta(\mathcal{Y})$ *is defined by*

$$h_{f,g}(\mathbf{x})[y] = \int_{\mathbb{R}^d} \mathbb{1}[f(\mathbf{x}+\mathbf{t}) = y] \cdot g(\mathbf{t})\, d\mathbf{t} \tag{3.6}$$

for all $y \in \mathcal{Y}$, and g is called the smoothing distribution*.*

Intuitively, f is the original model, and the value of the smoothed model $h_{f,g}$ at $\mathbf{x}$ is found by querying f on points around $\mathbf{x}$. We choose the points around $\mathbf{x}$ according to the distribution g, and the output $h_{f,g}(\mathbf{x})$ of the smoothed model is a probability distribution of the values of f at the queried points. In reasoning about the individual fairness of $h_{f,g}$, I will use the total variation distance (Definition 3.6) to measure the distance $D_{\Delta(\mathcal{Y})}$ between probability distributions.

Definition 3.6 (Total variation distance)**.** *Let $Y_1, Y_2 \in \Delta(\mathcal{Y})$. The* total variation distance *between Y_1 and Y_2 is*

$$D_{\Delta(\mathcal{Y})}(Y_1, Y_2) = \frac{1}{2} \sum_{y \in \mathcal{Y}} |Y_1[y] - Y_2[y]|.$$

3.3.2 Laplace Smoothing Distribution

One main difference between this setting and that in Section 3.2 is that we have a choice of the smoothing distribution g. Thus, instead of simply finding a metric under which the model is individually fair, we adapt the smoothing distribution to a given metric. Theorem 3.5 shows that, for any weighted L^p metric $D_{\mathcal{X}}$, there exists a smoothing distribution g that guarantees that $h_{f,g}$ is individually fair under $D_{\mathcal{X}}$ for all f.

Theorem 3.5 (Laplace smoothing)**.** *Let $\mathcal{M}_L$ be the set of all weighted L^p metrics. For any $D_{\mathcal{X}} \in \mathcal{M}_L$, let $g(\mathbf{t}) = \exp(-2D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}))/Z$, where Z is the normalization factor $\int_{\mathbb{R}^d} \exp(-2D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}))\, d\mathbf{t}$. Then, $h_{f,g}$ is individually fair under $D_{\mathcal{X}}$ for all f.*

Proof. I will show that $D_{\Delta(\mathcal{Y})}(h_{f,g}(\mathbf{x}), h_{f,g}(\mathbf{x}+\boldsymbol{\epsilon})) \leq D_{\mathcal{X}}(\mathbf{x}, \mathbf{x}+\boldsymbol{\epsilon})$ for all $\mathbf{x}, \boldsymbol{\epsilon} \in \mathbb{R}^d$.

First, since $D_{\mathcal{X}}(\mathbf{t}, \mathbf{t}-\boldsymbol{\epsilon}) = D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon})$ for all weighted L^p metric $D_{\mathcal{X}}$, we have

$$D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}) - D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon}) \leq D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}-\boldsymbol{\epsilon}) \leq D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}) + D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon}) \tag{3.7}$$

by the triangle inequality. We can apply the first inequality in Eq. (3.7) to bound the probability $g(\mathbf{t}-\boldsymbol{\epsilon})$ in terms of $g(\mathbf{t})$.

$$\begin{aligned} g(\mathbf{t}-\boldsymbol{\epsilon}) &= \exp(-2D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}-\boldsymbol{\epsilon}))/Z \\ &\leq \exp(-2[D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}) - D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon})])/Z \\ &= \exp(-2D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}))/Z \cdot \exp(2D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon})) \\ &= g(\mathbf{t}) \cdot \exp(2D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon})) \end{aligned}$$

[4] I will abuse notation and write g to denote both the distribution and its probability density function.

Then, for all $y \in \mathcal{Y}$ we have

$$\begin{aligned} h_{f,g}(\mathbf{x}+\boldsymbol{\epsilon})[y] &= \int_{\mathbb{R}^d} \mathbb{1}[f(\mathbf{x}+\boldsymbol{\epsilon}+\mathbf{t}) = y] \cdot g(\mathbf{t})\, d\mathbf{t} \\ &= \int_{\mathbb{R}^d} \mathbb{1}[f(\mathbf{x}+\mathbf{t}) = y] \cdot g(\mathbf{t}-\boldsymbol{\epsilon})\, d\mathbf{t} \\ &\leq \int_{\mathbb{R}^d} \mathbb{1}[f(\mathbf{x}+\mathbf{t}) = y] \cdot g(\mathbf{t}) \cdot \exp(2D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon}))\, d\mathbf{t} \\ &= h_{f,g}(\mathbf{x})[y] \cdot \exp(2D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon})). \end{aligned} \tag{3.8}$$

Similarly, we can apply the second inequality in Eq. (3.7) to derive the upper bound

$$h_{f,g}(\mathbf{x}+\boldsymbol{\epsilon})[y] \geq h_{f,g}(\mathbf{x})[y] / \exp(2D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon})). \tag{3.9}$$

We can now apply a previous result by Kairouz et al. [54, Theorem 6] to determine the maximum distance between $h_{f,g}(\mathbf{x})$ and $h_{f,g}(\mathbf{x}+\boldsymbol{\epsilon})$ that is attainable with the above constraints. For brevity, let c denote $\exp(2D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon}))$. In the context of ε-local differential privacy, Kairouz et al. showed that the maximum possible total variation distance is $(e^{\varepsilon}-1)/(e^{\varepsilon}+1)$. Replacing e^{ε} with c, we see that the distance between $h_{f,g}(\mathbf{x})$ and $h_{f,g}(\mathbf{x}+\boldsymbol{\epsilon})$ is at most $(c-1)/(c+1)$.

Finally, it remains to be proven that this quantity is not more than $D_{\mathcal{X}}(\mathbf{x}, \mathbf{x}+\boldsymbol{\epsilon})$, which is equivalent to $D_{\mathcal{X}}(\mathbf{0}, \boldsymbol{\epsilon})$ for weighted L^p metrics. Since this distance can be written as $(\ln c)/2$, it suffices to show that $(c-1)/(c+1) \leq (\ln c)/2$ for all $c \geq 1$. This inequality follows from the fact that equality holds at $c = 1$ and that the derivative of the right-hand side is never less than that of the left-hand side for $c \geq 1$. □

Although Theorem 3.5 identifies a smoothing distribution that ensures the individual *fairness* of the resulting model under $D_{\mathcal{X}}$, we also want the smoothed model to retain the *utility* of the original model f. In the extreme case where g is the uniform distribution over $\mathbb{R}^d$, the resulting model will be a constant function and therefore satisfy individual fairness under any metric, but it will not be very useful for classification tasks. More generally, smoothed models that are individually fair under smaller distance metrics tend to not preserve as much locally relevant information about f. Thus, I argue that a smoothing distribution does not unnecessarily lower the model's utility by showing that $D_{\mathcal{X}}$ is *minimal*. The definition of minimality that I use here differs from Definition 3.3 in that the smoothed model must be individually fair for all f.

Definition 3.7 (Minimal distance metric, smoothing)**.** *Let $\mathcal{M} \subseteq \mathcal{M}_L$ be a set of distance metrics in $\mathbb{R}^d$. A metric $D_{\mathcal{X}} \in \mathcal{M}$ is* minimal *in $\mathcal{M}$ with respect to a smoothing distribution g if* (1) *$h_{f,g}$ is individually fair under $D_{\mathcal{X}}$ for all f, and* (2) *there does not exist a different $D'_{\mathcal{X}} \in \mathcal{M}$ such that $h_{f,g}$ is individually fair under $D'_{\mathcal{X}}$ for all f and $D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) \geq D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2)$ for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$.*

For general weighted L^p metrics, the inequalities in Eq. (3.7) are strict for most $\mathbf{t}$, so the bounds in Eqs. (3.8) and (3.9) are not tight, and $D_{\mathcal{X}}$ is not guaranteed to be minimal. On the other hand, if $D_{\mathcal{X}}$ is a weighted L^1 metric, Theorem 3.6 shows that it is minimal with respect to its Laplace smoothing distribution.

Theorem 3.6. *Let $\mathcal{M}_1$ be the set of all weighted L^1 metrics. For any $D_{\mathcal{X}} \in \mathcal{M}_1$, let $g(\mathbf{t}) = \exp(-2D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}))/Z$, where Z is the normalization factor $\int_{\mathbb{R}^d} \exp(-2D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}))\, d\mathbf{t}$. Then, $D_{\mathcal{X}}$ is uniquely minimal in $\mathcal{M}_1$ with respect to g.*

Proof. I have already proven in Theorem 3.5 that $h_{f,g}$ is individually fair under $D_{\mathcal{X}}$ for all f. It remains to show that there does not exist a different $D'_{\mathcal{X}} \in \mathcal{M}_1$ such that $h_{f,g}$ is individually fair under $D'_{\mathcal{X}}$ for all f and $D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) \geq D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2)$ for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$.

Let $w_1, \ldots, w_d$ and $w'_1, \ldots, w'_d$ be the weights of $D_{\mathcal{X}}$ and $D'_{\mathcal{X}}$, respectively. If $D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2) \geq D'_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2)$ for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$, we must have $w_i \geq w'_i$ for all i. Moreover, since $D_{\mathcal{X}} \neq D'_{\mathcal{X}}$, there exists i such that $w_i > w'_i$. We now construct f such that $h_{f,g}$ is not individually fair under $D'_{\mathcal{X}}$.

Let $f : \mathbb{R}^d \to \{0, 1\}$ be a function such that $f(\mathbf{x}) = \mathbb{1}[x_i \geq 0]$. I will show that there exists $\epsilon > 0$ such that $D_{\Delta(\mathcal{Y})}(h_{f,g}(\mathbf{0}), h_{f,g}(\epsilon \mathbf{e}_i)) > D'_{\mathcal{X}}(\mathbf{0}, \epsilon \mathbf{e}_i)$, where $\mathbf{e}_i$ is the basis vector that is one in the i-th coordinate and zero in all others. Applying Eq. (3.6) and simplifying, we get $h_{f,g}(\mathbf{0})[0] = 1/2$ and $h_{f,g}(\epsilon \mathbf{e}_i)[0] = \exp(-2w_i\epsilon)/2$. Therefore, the distance $D_{\Delta(\mathcal{Y})}$ is $(1 - \exp(-2w_i\epsilon))/2$. Moreover, we have $D'_{\mathcal{X}} = w'_i\epsilon$. The ratio $D_{\Delta(\mathcal{Y})}/D'_{\mathcal{X}}$ approaches $w_i/w'_i > 1$ as $\epsilon \to 0$, so when ϵ is sufficiently small, we have $D_{\Delta(\mathcal{Y})} > D'_{\mathcal{X}}$.

Uniqueness follows from the argument given in the last paragraph of the proof of Theorem 3.1. □

3.3.3 Gaussian Smoothing Distribution

As I will show in Section 3.5, in practice Laplace smoothing distributions do not preserve well the utility of f due to their relatively high densities at the tails. Thus, I present Gaussian smoothing as an alternative, which Theorem 3.7 shows is individually fair under any weighted L^2 metric. Since $D_2(\mathbf{x}_1, \mathbf{x}_2) \leq d^{\max(0,1/2-1/p)} D_p(\mathbf{x}_1, \mathbf{x}_2)$ for any weighted L^2 and L^p metrics D_2 and D_p with the same weights, we can then scale the weights accordingly to make $h_{f,g}$ fair under any given L^p metric. For simplicity, we only consider the setting of binary classification, i.e., $\mathcal{Y} = \{0, 1\}$.

Theorem 3.7 (Gaussian smoothing)**.** *Let $D_{\mathcal{X}}$ be a weighted L^2 metric with weights $w_1, \ldots, w_d$, and let $\boldsymbol{\Sigma}$ be a diagonal matrix with $\Sigma_{ii} = (2\pi w_i)^{-1}$. If g is Gaussian with mean $\mathbf{0}$ and variance $\boldsymbol{\Sigma}$, $h_{f,g}$ is individually fair under $D_{\mathcal{X}}$ for all $f : \mathbb{R}^d \to \{0, 1\}$.*

To prove this theorem, I will apply the Neyman–Pearson lemma [73], as formulated by Cohen et al. [25, Lemma 3].

Lemma 3.8 (Neyman–Pearson)**.** *Let X_1 and X_2 be random variables in $\mathbb{R}^d$ with densities μ_{X_1} and μ_{X_2}, and let $f, f^\star : \mathbb{R}^d \to \{0, 1\}$ such that $f^\star(\mathbf{t}) = 1$ if and only if $\mu_{X_2}(\mathbf{t})/\mu_{X_1}(\mathbf{t}) \geq k$ for some threshold $k > 0$. Then,*

$$\Pr[f(X_1) = 1] = \Pr[f^\star(X_1) = 1] \textit{ implies } \Pr[f(X_2) = 1] \leq \Pr[f^\star(X_2) = 1].$$

Proof of Theorem 3.7. I proceed by showing that $D_{\Delta(\mathcal{Y})}(h_{f,g}(\mathbf{x}), h_{f,g}(\mathbf{x} + \boldsymbol{\epsilon})) \leq D_{\mathcal{X}}(\mathbf{x}, \mathbf{x} + \boldsymbol{\epsilon})$ for all $\mathbf{x}, \boldsymbol{\epsilon} \in \mathbb{R}^d$ and $f : \mathbb{R}^d \to \{0, 1\}$. For any given f, we will first find $f^\star$ such that

$$D_{\Delta(\mathcal{Y})}(h_{f,g}(\mathbf{x}), h_{f,g}(\mathbf{x} + \boldsymbol{\epsilon})) \leq D_{\Delta(\mathcal{Y})}(h_{f^\star,g}(\mathbf{x}), h_{f^\star,g}(\mathbf{x} + \boldsymbol{\epsilon})). \tag{3.10}$$

I will then show that $h_{f^\star,g}$ is individually fair under $D_{\mathcal{X}}$, which together with Eq. (3.10) implies that $h_{f,g}$ is also individually fair under $D_{\mathcal{X}}$.

Fix $\mathbf{x}, \boldsymbol{\epsilon} \in \mathbb{R}^d$, and assume without loss of generality that $h_{f,g}(\mathbf{x})[1] \leq h_{f,g}(\mathbf{x} + \boldsymbol{\epsilon})[1]$. Then, we have

$$D_{\Delta(\mathcal{Y})}(h_{f,g}(\mathbf{x}), h_{f,g}(\mathbf{x} + \boldsymbol{\epsilon})) = h_{f,g}(\mathbf{x} + \boldsymbol{\epsilon})[1] - h_{f,g}(\mathbf{x})[1]. \tag{3.11}$$

We apply Lemma 3.8 by choosing X_1 and X_2 such that $g(\mathbf{t}) = \mu_{X_1}(\mathbf{x}+\mathbf{t}) = \mu_{X_2}(\mathbf{x}+\boldsymbol{\epsilon}+\mathbf{t})$. By Eq. (3.6), we have $\Pr[f(X_1) = 1] = h_{f,g}(\mathbf{x})[1]$ and $\Pr[f(X_2) = 1] = h_{f,g}(\mathbf{x}+\boldsymbol{\epsilon})[1]$, and similar relations hold between $f^\star$ and $h_{f^\star,g}$. Therefore, if there exists $f^\star$ that satisfies the condition in Lemma 3.8 such that $h_{f,g}(\mathbf{x})[1] = h_{f^\star,g}(\mathbf{x})[1]$, then $h_{f,g}(\mathbf{x}+\boldsymbol{\epsilon})[1] \leq h_{f^\star,g}(\mathbf{x}+\boldsymbol{\epsilon})[1]$. Combining these two (in)equalities, we get

$$h_{f,g}(\mathbf{x}+\boldsymbol{\epsilon})[1] - h_{f,g}(\mathbf{x})[1] \leq h_{f^\star,g}(\mathbf{x}+\boldsymbol{\epsilon})[1] - h_{f^\star,g}(\mathbf{x})[1],$$

and Eq. (3.10) follows from Eq. (3.11) and its $h_{f^\star,g}$ counterpart.

I now show that it is possible to find $f^\star$ such that $h_{f^\star,g}(\mathbf{x})[1] = h_{f,g}(\mathbf{x})[1]$. By construction, we have that $f^\star(\mathbf{t}) = 1$ if and only if

$$\frac{\mu_{X_2}(\mathbf{t})}{\mu_{X_1}(\mathbf{t})} = \frac{g(\mathbf{t}-\mathbf{x}-\boldsymbol{\epsilon})}{g(\mathbf{t}-\mathbf{x})} \geq k$$

for some $k > 0$. Substituting in the Gaussian density function and solving for $\mathbf{t}$, we see that this inequality holds whenever $\boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\mathbf{t} \geq \kappa$, where κ is a constant with respect to $\mathbf{t}$. When evaluating $h_{f^\star,g}(\mathbf{x})$ as per Eq. (3.6), $\mathbf{t}$ is distributed normally, and therefore $\boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\mathbf{t}$ is also (univariate) Gaussian. Thus, with the appropriate value of κ we can obtain the desired $f^\star$.

Finally, it remains to show that $h_{f^\star,g}$ is individually fair under $D_{\mathcal{X}}$. Let $\tau = \boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\mathbf{t}$, and let γ be the density function of τ. With some computation, we see that $f^\star(\mathbf{t})$ and $f^\star(\mathbf{t}+\boldsymbol{\epsilon})$ differ if and only if $\kappa \leq \tau < \kappa + \boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}$. Moreover, since $\mathbf{t}$ has variance $\boldsymbol{\Sigma}$, the variance of $\tau = \boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\mathbf{t}$ is $\boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\mathrm{Var}(\mathbf{t})(\boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1})^T = \boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}$, and thus the maximum value of γ is $(2\pi\boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon})^{-1/2}$. We apply these two facts to arrive at the desired result:

$$\begin{aligned}
D_{\Delta(\mathcal{Y})}(h_{f^\star,g}(\mathbf{x}), h_{f^\star,g}(\mathbf{x}+\boldsymbol{\epsilon})) &= h_{f^\star,g}(\mathbf{x}+\boldsymbol{\epsilon})[1] - h_{f^\star,g}(\mathbf{x})[1] \\
&= \int_{\mathbb{R}^d} (f^\star(\mathbf{x}+\boldsymbol{\epsilon}+\mathbf{t}) - f^\star(\mathbf{x}+\mathbf{t})) \cdot g(\mathbf{t})\, d\mathbf{t} \\
&= \int_{\mathbb{R}^d} (f^\star(\mathbf{t}+\boldsymbol{\epsilon}) - f^\star(\mathbf{t})) \cdot g(\mathbf{t}-\mathbf{x})\, d\mathbf{t} \\
&= \int_{\kappa}^{\kappa+\boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}} \gamma(\tau - \boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\mathbf{x})\, d\tau \\
&\leq \boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon} \cdot (2\pi\boldsymbol{\epsilon}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon})^{-1/2} \\
&= \left(\sum_{i=1}^{d} \epsilon_i^2/(2\pi\Sigma_{ii})\right)^{1/2} \\
&= \left(\sum_{i=1}^{d} \epsilon_i^2 \cdot w_i\right)^{1/2} \\
&= D_{\mathcal{X}}(\mathbf{x}, \mathbf{x}+\boldsymbol{\epsilon}).
\end{aligned}$$

□

I end this section with Theorem 3.9, which states that an L^2 metric $D_{\mathcal{X}}$ is minimal with respect to its Gaussian smoothing distribution. The proof is omitted since it is very similar to that of Theorem 3.6.

Algorithm 3.1 Randomized smoothing by sampling

Require: Model $f : \mathbb{R}^d \to \mathcal{Y}$, point $\mathbf{x} \in \mathbb{R}^d$, parameters $p \in \{1, 2, \infty\}$, $\boldsymbol{w} \in \mathbb{R}^d_+$ for the weighted L^p metric $D_{\mathcal{X}}$

Ensure: Distribution in $\Delta(\mathcal{Y})$ that approximates $h_{f,g}(\mathbf{x})$

```
//Initialize the output probability distribution.
for y ∈ 𝒴 do
    prob[y] ← 0
end for
//Evaluate f at n randomly sampled points around x.
for j = 1, ..., n do
    t_j ← sampleNoise(d, p, w)
    y_j ← f(x + t_j)
    prob[y_j] ← prob[y_j] + 1/n
end for
return prob
```

Theorem 3.9. *Let $\mathcal{M}_2$ be the set of all weighted L^2 metrics. For any $D_{\mathcal{X}} \in \mathcal{M}_2$, let $w_1, \ldots, w_d$ be the weights, and let $\boldsymbol{\Sigma}$ be a diagonal matrix with $\Sigma_{ii} = (2\pi w_i)^{-1}$. If g is a Gaussian with mean $\mathbf{0}$ and variance $\boldsymbol{\Sigma}$, $D_{\mathcal{X}}$ is uniquely minimal in $\mathcal{M}_2$ with respect to g.*

3.4 Practical Implementation

In practice, it is infeasible to compute $h_{f,g}(\mathbf{x})$ because of the integral in Eq. (3.6). Therefore, to apply randomized smoothing in practice, I approximate the integral with Algorithm 3.1, i.e., by sampling n points independently from the smoothing distribution g, evaluating the model with this noise added to $\mathbf{x}$, and returning the observed probability of predicting each class on the sampled points. However, the resulting model may not be individually fair due to the finite sample size. Thus, I define and prove (ϵ, δ)-individual fairness, which requires that the model be close to individually fair with high probability.

Definition 3.8 ((ϵ, δ)-individual fairness)**.** *A randomized model $h : \mathcal{X} \to \Delta(\mathcal{Y})$ is* (ϵ, δ)-individually fair *under metric $D_{\mathcal{X}} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ if, for all $x_1, x_2 \in \mathcal{X}$,*

$$D_{\Delta(\mathcal{Y})}(h(x_1), h(x_2)) \leq D_{\mathcal{X}}(x_1, x_2) + \epsilon \tag{3.12}$$

with probability at least $1 - \delta$. The probability is taken over the randomness of h.

Theorem 3.10. *Let $h^n_{f,g}$ be a model that approximates $h_{f,g}$ with n samples. If $|\mathcal{Y}| = m$ and $h_{f,g}$ is fair under $D_{\mathcal{X}}$, then $h^n_{f,g}$ is $(\epsilon, 2me^{-4n\epsilon^2/m^2})$-individually fair under $D_{\mathcal{X}}$.*

Proof. Consider any two points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$. Since $h_{f,g}$ is individually fair under $D_{\mathcal{X}}$, we have

$$\tfrac{1}{2} \textstyle\sum_{y \in \mathcal{Y}} |h_{f,g}(\mathbf{x}_1)[y] - h_{f,g}(\mathbf{x}_2)[y]| = D_{\Delta(\mathcal{Y})}(h_{f,g}(\mathbf{x}_1), h_{f,g}(\mathbf{x}_2)) \leq D_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2).$$

I will show that $|\text{diff}[y]| > 2\epsilon/m$ with probability less than $2e^{-4n\epsilon^2/m^2}$, where $\text{diff}[y]$ is shorthand for $(h^n_{f,g}(\mathbf{x}_1)[y] - h^n_{f,g}(\mathbf{x}_2)[y]) - (h_{f,g}(\mathbf{x}_1)[y] - h_{f,g}(\mathbf{x}_2)[y])$. Then, by union bound, with probability at least $1 - 2me^{-4n\epsilon^2/m^2}$ we will have $\text{diff}[y] \leq 2\epsilon/m$ for all $y \in \mathcal{Y}$, which leads to our desired result.

$$\begin{aligned} D_{\Delta(\mathcal{Y})}(h^n_{f,g}(\mathbf{x}_1), h^n_{f,g}(\mathbf{x}_2)) &= \tfrac{1}{2}\textstyle\sum_{y\in\mathcal{Y}} |h^n_{f,g}(\mathbf{x}_1)[y] - h^n_{f,g}(\mathbf{x}_2)[y]| \\ &\leq \tfrac{1}{2}\textstyle\sum_{y\in\mathcal{Y}} |h_{f,g}(\mathbf{x}_1)[y] - h_{f,g}(\mathbf{x}_2)[y]| + \tfrac{1}{2}\textstyle\sum_{y\in\mathcal{Y}} |\text{diff}[y]| \\ &\leq D_{\mathcal{X}}(x_1, x_2) + \epsilon \end{aligned}$$

Fix $y \in \mathcal{Y}$, and let $X_{ij} = \mathbb{1}[f(\mathbf{x}_i + \mathbf{t}_{ij}) = y]$, where $\mathbf{t}_{ij}$ is the j-th sample drawn from the smoothing distribution g while evaluating $h^n_{f,g}(\mathbf{x}_i)$. Then, $h^n_{f,g}(\mathbf{x}_i)[y] = \frac{1}{n}\sum_{j=1}^n X_{ij}$ and $h_{f,g}(\mathbf{x}_i)[y] = \frac{1}{n}\mathbb{E}[\sum_{j=1}^n X_{ij}]$, so $\text{diff} = \frac{1}{n}\sum_{j=1}^n (X_{1j} - X_{2j} - \mathbb{E}[X_{1j} - X_{2j}])$. The theorem follows from Hoeffding's inequality.

$$\Pr[|\text{diff}[y]| > 2\epsilon/m] = \Pr[|\tfrac{1}{2n}\textstyle\sum_{j=1}^n (X_{1j} - X_{2j} - \mathbb{E}[X_{1j} - X_{2j}])| > \epsilon/m] < 2e^{-4n\epsilon^2/m^2} \quad \square$$

3.4.1 Noise Sampling

Implementations of Gaussian noise sampling are commonly included in data analysis libraries. For Laplace noise sampling, we can apply Algorithm 3.2, which describes how to sample a point $\mathbf{t}$ from the Laplace smoothing distribution when $p \in \{1, 2, \infty\}$. Without loss of generality, we assume that $D_{\mathcal{X}}$ is a standard L^p metric since we can simply rescale each coordinate by its weight w_i. Recall that $g(\mathbf{t}) \propto \exp(-2D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}))$. When $p = 1$, this quantity becomes $\exp(-2\sum_{i=1}^d w_i \cdot |t_i|) = \prod_{i=1}^d \exp(-2w_i \cdot |t_i|)$, so each coordinate can be sampled from the Laplace distribution independently of the others. For other values of p, the coordinates are not independent, so we instead sample the distance $r = D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}) = \|\mathbf{t}\|_p$ and then pick a point $\mathbf{t}$ uniformly at random on the sphere ($p = 2$) or hypercube ($p = \infty$) of radius r.

To sample r, note that the set $\{\mathbf{t} \mid \|\mathbf{t}\|_p = r\}$ has surface area proportional to r^{d-1}. Hence, the probability of drawing a point in this set from the distribution g is proportional to $r^{d-1}e^{-2r}$, and the cumulative distribution function of the radius r is $P(d, 2r)$, where P is the regularized lower incomplete gamma function. Finally, computing the inverse of this function allows us to sample r through inverse transform sampling.

3.5 Experiments

Theorems 3.5, 3.7, and 3.10 show that smoothed models created using Algorithm 3.1 are individually fair. By contrast, we have no similar results about their utility except heuristic arguments from minimality. Thus, in this section I measure the utility of smoothed models $h_{f,g}$ on four real-world datasets (detailed below), using the smoothing distributions described in Theorems 3.5 and 3.7.

For each dataset, I first trained a logistic regression model, and then the weights of $D_{\mathcal{X}}$ were chosen to be a quarter of the regression coefficients, in accordance with Theorem 3.4. In particular, this means that the features that receive little weight in the linear model, which are

Algorithm 3.2 Laplace noise sampling

Require: Positive integer d, parameters $p \in \{1, 2, \infty\}$, $\boldsymbol{w} \in \mathbb{R}^d_+$ for the weighted L^p metric $D_{\mathcal{X}}$
Ensure: Noise $\mathbf{t} \in \mathbb{R}^d$ drawn randomly from distribution g such that $g(\mathbf{t}) \propto \exp(-2D_{\mathcal{X}}(\mathbf{0}, \mathbf{t}))$

```
//Sample noise under the standard L^p metric.
if p = 1 then
  //Each coordinate is independent when p = 1.
  for i = 1, ..., d do
    t_i ~ Laplace(0, 0.5)
  end for
else
  //Sample a random point on the unit L^p-sphere.
  for i = 1, ..., d do
    if p = 2 then
      t_i ~ Gaussian(0, 1)
    else if p = ∞ then
      t_i ~ Uniform(−1, 1)
    end if
  end for
  t ← t/‖t‖_p
  //Use inverse transform sampling for radius r = ‖t‖_p.
  u ~ Uniform(0, 1)
  r ← P^{-1}(d, u)/2 {P is regularized lower incomplete gamma function}
  t ← r · t
end if
//Adjust the noise, taking the weights into account.
for i = 1, ..., d do
  t_i ← t_i/w_i
end for
return (t_1, ..., t_d)
```

thus less likely to be predictive, have little effect on the output of the smoothed model. After that, for each dataset I trained a neural network f with two dense hidden layers of 128 ReLU neurons each, and I evaluated the utility of both Laplace- and Gaussian-smoothed models. When training the neural networks, I augmented the training data with noise drawn from the smoothing distribution, as prior work [25] shows that this improves the utility of the smoothed model. At test time, for each point to be evaluated, I randomly sampled $n = 10^5$ points according to the smoothing distribution. By Theorem 3.10, this corresponds to a guarantee of $\delta = 1.8 \times 10^{-4}$ at $\epsilon = 10^{-2}$.

Adult. My model uses the five numerical features from the UCI Adult dataset [33] to predict whether a person earns more than $50,000 per year.

COMPAS. I use the dataset compiled by ProPublica [6] to analyze the COMPAS recidivism prediction model [36]. My model uses eight features (15 when one-hot encoded) to predict whether

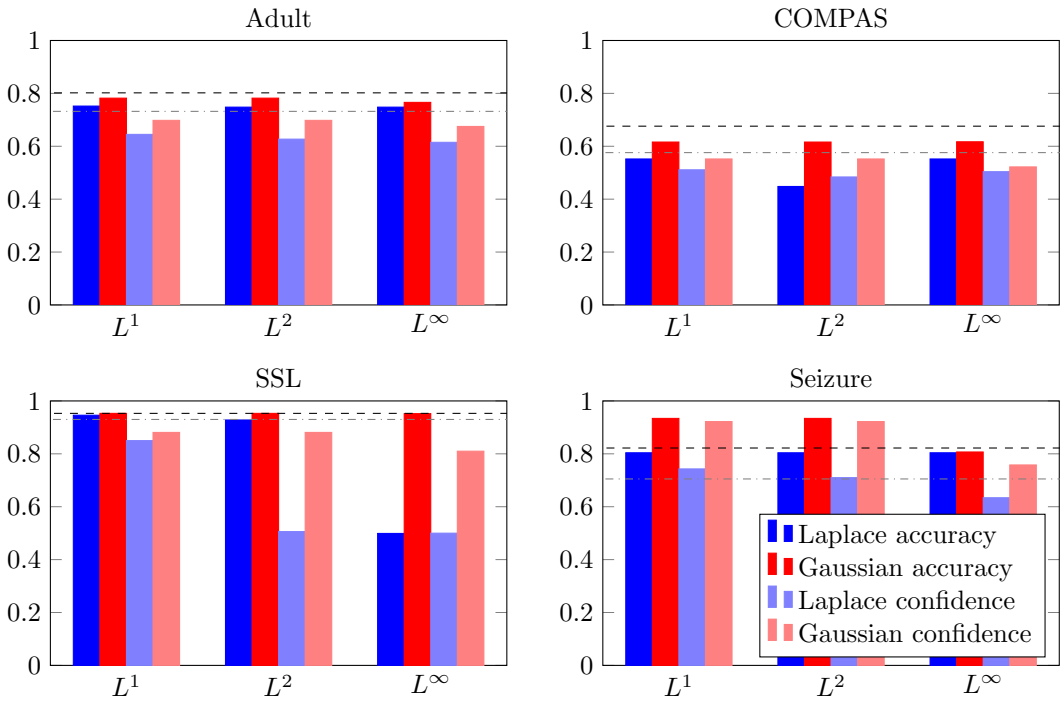


Figure 3.1: Utility of smoothed models derived from the four datasets described in Section 3.5. The black dashed line indicates the accuracy of the logistic regression model, and the dash-dotted line its average confidence in the correct label. Because smoothed models output probabilities, accuracy and mean confidence are both reasonable measures of their utility, but only mean confidence preserves the individual fairness of the smoothed model.

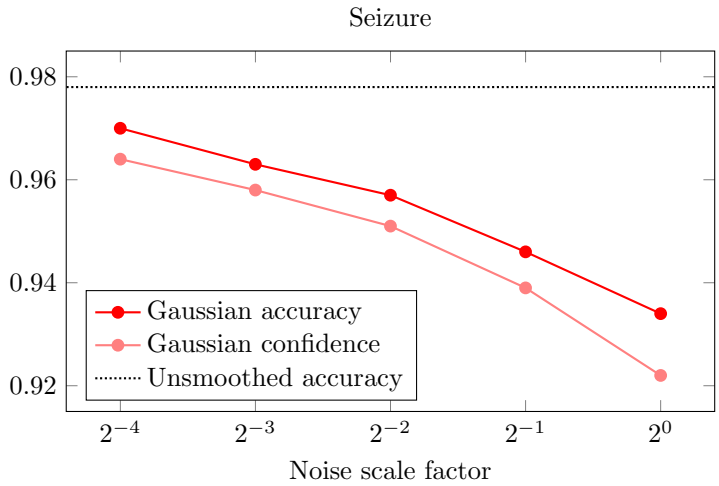


Figure 3.2: Fairness-utility trade-off for randomized smoothing on the Seizure dataset for L^1 and L^2 metrics. The horizontal axis indicates the magnitude of the added noise as a fraction of what is necessary to achieve the same individual fairness guarantees as the logistic regression model trained on the same dataset. Larger noise gives stronger fairness guarantees at the cost of reduced utility.

a person will recidivate within the next two years.
SSL. The Strategic Subject List dataset [24] contains scores given by Chicago Police Department's model to rate a person's risk of being involved in a shooting incident, either as a perpetrator or a victim. My model uses the same eight numerical features used by Chicago's model to predict a person's SSL risk score.
Seizure. In the UCI Epileptic Seizure dataset [5, 33], every row consists of 178 readings from an EEG taken over a second. My model predicts whether a person is experiencing a seizure during that second.

3.5.1 Results

I applied both Laplace and Gaussian smoothing to create models that are individually fair under the weighted L^1, L^2, or L^∞ metrics, with weights derived from those of the corresponding logistic regression model as previously described. Because the outputs of the smoothed models are probabilities, I measured their utilities both in terms of standard accuracy and the mean confidence assigned to the correct class, $\mathbb{E}_{(\mathbf{x},y)}[h_{f,g}(\mathbf{x})[y]]$. Although accuracy is a more common measure of utility, its use of the thresholding operator argmax is incompatible with the individual fairness of $h_{f,g}$.

The results in Fig. 3.1 show that Gaussian-smoothed models approximately match or exceed the performance of the logistic model while achieving the same individual fairness guarantees. The three datasets other than Seizure contain mostly linear relationships, so the unsmoothed neural networks had were not much more accurate (within 1.5 percentage points) than the logistic regression models. As a result, the smoothed neural networks tended to perform slightly worse than the logistic regression models. By contrast, the Seizure dataset benefits from the use of a neural network, which achieved 97.8% accuracy. As a result, the Gaussian-smoothed Seizure model had an accuracy of 93.4% and confidence of 92.2%, far exceeding those of the logistic model (82.2% accuracy and 70.5% confidence).

Although L^1 metrics are minimal with respect to Laplace smoothing, Gaussian smoooothing outperforms on these metrics in practice. Laplace distributions have higher densities at the tails, resulting in more queries that are very dissimilar to the input point $\mathbf{x}$. Thus, in practice I recommend using Gaussian smoothing for every L^p metric, adjusting the weights as shown in Section 3.3.3 to account for the value of p. Following my own recommendation, for the next set of experiments I only evaluated Gaussian smoothing distributions.

Fig. 3.2 details the effect of varying the parameters of the smoothing distribution. As expected, if we relax the fairness guarantees by decreasing the magnitude of the added noise, the accuracy and confidence of the resulting model both increase. In particular, when the target similarity metric $D_{\mathcal{X}}$ in Theorem 3.7 is 16 times as large as the minimal metric of the logistic regression model, the added noise is 16 times as small, and the accuracy and confidence of the resulting model are 97.0% and 96.4%, respectively. This contrasts with the 93.4% accuracy and 92.2% confidence for a model with the same fairness guarantees as the logistic regression model. Therefore, in some settings, it may be desirable to trade off some fairness guarantees in order to create a more predictive model.

Chapter 4

Auditing for Distribution Shifts

I now explore the issue that a model can have varying accuracy on different protected groups. In some cases an incorrect prediction can cause more harm than receiving a generally unfavorable outcome. For example, the accuracy disparity example in Chapter 1 shows that, for loan applicants who would end up defaulting on the loan, it can be a better outcome to be denied for the loan. Therefore, if a model that is disproportionately inaccurate on one protected group, its effect can still be discriminatory even if it satisfies statistical parity.

The issue of accuracy disparity has previously been studied by Zafar et al. [109], who call it "disparate mistreatment". This issue is not merely theoretical—Buolamwini and Gebru [15] demonstrated that real-world commercial facial recognition models are significantly less accurate on darker-skinned and female faces. They also show that darker-skinned and female faces are underrepresented in existing benchmark datasets used for evaluating models. Therefore, it is plausible that the training sets of the commercial models also suffer from a similar underrepresentation issue that at least partially causes the disparity in accuracy. However, Buolamwini and Gebru do not draw any conclusions to this effect, and it is difficult for the general public to analyze the training sets of the commercial models because they tend to be proprietary.

At the same time, machine learning models are increasingly used to make consequential decisions about people, such as college admissions, loan approvals, and hiring. This means that the public has a stake in correcting any accuracy disparity that may arise in these models. However, we cannot automatically assume that an accuracy disparity is evidence of underrepresentation, as the choice of a learning algorithm [46] or features [20] can also lead to an accuracy disparity. Still, it is important to identify the cause of an accuracy disparity, as the remedy for the disparity depends on the cause.

Motivated by this concern, I present a black-box audit that checks for a *distribution shift* (DS), which occurs when the data distribution to which a model is applied differs from the model's training distribution. Unrepresentative training sets characterize a special case of DS that we call a *group distribution shift* (GDS). In Section 4.2, I formally define the audit as a game between the auditor and the entity holding the model. Section 4.3 details a novel black-box DS auditing procedure based on shadow models, a technique previously used for membership inference attacks [87]. In Section 4.3.3, I theoretically show how the overfitting of a model can leak information about the model's disparate behavior across groups. This insight is then used to develop a GDS audit that can detect underrepresentation issues in the training distribution.

Finally, I empirically evaluate the audit following the experimental methodology described in Section 4.4. The results in Section 4.5 show that the audit achieves near-perfect accuracy for DS and 80–100% AUC-ROC for GDS.

The contents of this chapter are based on a work currently under submission at the ACM Conference on Computer and Communications Security [52].

4.1 Related Work

The term "dataset shift" was used for the first time in a 2009 book compiling seminal works on the field [79]. Much of these works focus on learning under distribution shift, which is closely related to other fields, such as *domain adaptation* [113], *multi-task learning* [114], and *transfer learning* [19, 102, 103].

Another line of work is to detect shifts and determine if adaptation is necessary. A post-hoc approach to detect DS is to monitor the error of the model during deployment. However, this approach is not well suited for production systems, which must maintain good model performance for the entire duration of its deployment. Instead, prior work assumes access to the training set and an unlabeled set of test samples, casting the problem as a two-sample hypothesis test between the training and test distributions[5] [66, 80].

When a sample of the test distribution is not available, out-of-distribution detection methods attempt to detect single test points that do not belong to the training distribution. I refer the reader to a review by Shafaei et al. [85] for an overview of these methods. With the increasing popularity of deep neural networks, which tend to be over-confident in their predictions [41], there has been a surge of out-of-distribution detection research specifically designed for deep neural networks [45, 63, 83].

Because all these works frame the problem from the model holder's perspective, and not from an adversary's point of view, they assume access to the training set. To the best of my knowledge, the work presented in this chapter is the first to detect shifts in an adversarial setting without access to the training set.

Membership and dataset inference. To achieve our goal of auditing a model without access to the model's training set, we rely on techniques used for membership inference attacks. In my work on membership inference [107, 108], I formally defined the membership inference attack, where the attacker seeks to determine, with only query access to a model, whether a given data point is in the model's training set. This definition is the basis for the formal definition of the DS audit presented in this chapter. The difference is that the goal is now to learn information about the entire training distribution rather than a specific point.

Although I previously attained a reasonably successful membership inference attack simply by leveraging the fact that models are often more accurate on their training sets, the attack presented by Shokri et al. [87] is even more successful due to its use of *shadow models*. Shadow models are trained to imitate the behavior of the target model, and this allows the attacker to

[5] In this section, the *test* distribution refers to that encountered during deployment, not the one that was used to evaluate the model during the development phase.

query the shadow models to learn how the target model may behave differently on its training set compared to the general population. The DS audit also uses shadow models, but their purpose is different. They illustrate how a model would behave if there is no DS, so by comparing them to the target model, the auditor can decide whether the target model's training set distribution is different from what it should be.

A recent study by Maini et al. [71] presents black-box methods to verify the ownership of a model. The methods exploit the fact that most learning algorithms tend to maximize the distance of the training points to the decision boundary. To verify whether a model has been trained on a proprietary training set, they probe the distance between points from the proprietary set to the model's decision boundary. Dataset inference is closely related to membership inference in that it checks the model for knowledge about specific data points. By contrast, the DS audit checks for differences between data distributions.

Social harms of distribution shifts. There has been extensive research on fair machine learning [11], and one aspect of fairness that has started to receive attention is accuracy disparity [110, 117]. This is not a merely theoretical issue—Buolamwini and Gebru [15] demonstrated that commercial facial recognition libraries are significantly less accurate on darker-skinned and female faces. Their study and others have shown that these faces are underrepresented in existing benchmark datasets used for evaluating face recognition models [15, 55]. This suggests that demographically imbalanced training sets may be at least in part responsible for the accuracy disparity.

Recent works have studied fairness with the lens of DS. Singh et al. [88] propose a technique to minimize the harm that DS can have on both prediction accuracy and fairness. Rawal et al. [81] review recourse actions that are recommended to individuals to recover from an unfair prediction, showing that the recourses obtained from existing techniques are not guaranteed to be valid under DS.

4.2 Problem Statement

In this section, I formally define the audit by specifying the types of distribution shift we are trying to detect. I first describe my definition of membership inference [107], and then I modify this definition to define the DS audit.

4.2.1 Background on Membership Inference

In a membership inference attack, the attacker is given a point that is drawn from either the training set of a model or the general population, and his goal is to figure out which distribution the point was drawn from. This is presented more formally as a game in Definition 4.1.

Definition 4.1 (Membership inference game)**.** *Let A be a learning algorithm, n be a positive integer, and $\mathcal{D}$ be a distribution over data points (x, y). The membership inference game proceeds as follows:*

1. *The challenger samples a training set $S \sim \mathcal{D}^n$.*

2. *The challenger trains model $h_S = A(S)$.*
3. *The challenger chooses $b \leftarrow \{0, 1\}$ uniformly at random.*
4. *The challenger draws $(x, y) \sim S$ if $b = 0$, and $(x, y) \sim \mathcal{D}$ if $b = 1$.*
5. *The attacker is given (x, y) and black-box access to h_S.*
    - *The attacker also knows A, n, and $\mathcal{D}$, which are assumed to be public information.*
6. *The attacker wins if he outputs the correct value of b.*

Here and afterwards, x represents the inputs to the model, and y is the value of the response or outcome variable.

4.2.2 Distribution Shift

To formalize the auditor's task in detecting DS, I propose a game-based definition similar to Definition 4.1. The new definition differs in one major way that reflects the change in the objective of the attacker, whom we now call the auditor. Because the objective is to figure out which distribution the training set was drawn from, the uniformly random bit b now decides the distribution from which to draw the training set. One possible distribution is $\mathcal{D}$, which is assumed to be public information known to the auditor. This represents the *normative* distribution that the model should be trained on. The other possibility is the *alternative* distribution $\mathcal{D}'$, which represents the distribution that the audited model was actually trained with. As a black-box auditor does not have access to the training set of the model being audited, we assume that $\mathcal{D}'$ is not known to the auditor.

Definition 4.2 (Distribution shift audit)**.** *Let A be a learning algorithm, n be a positive integer, and $\mathcal{D}$ and $\mathcal{D}'$ be distributions over data points (x, y). The distribution shift audit game proceeds as follows:*

1. *The challenger chooses $b \leftarrow \{0, 1\}$ uniformly at random.*
2. *The challenger samples training set $S \sim \mathcal{D}^n$ if $b = 0$, and $S \sim (\mathcal{D}')^n$ if $b = 1$.*
3. *The challenger trains model $h_S = A(S)$.*
4. *The auditor is given black-box access to h_S.*
    - *The auditor also knows A, n, and $\mathcal{D}$, which are assumed to be public information.*
5. *The auditor wins if she outputs the correct value of b.*

Threat model. As stated in Definition 4.2, we assume a black-box audit, i.e., the auditor can query the audited model and observe its output on any input of her choice, but she cannot observe the model weights and intermediate computations. The auditor also knows A, the learning algorithm used to train the model, and its parameters. This assumption is common in the literature on adversarial machine learning [87]. Moreover, companies that give query access to commercial models often publish details about the models but not their data [69, 93, 94]. We also assume that the auditor knows the size of the training set, n. Section 4.5.3 shows that in practice the auditor does not need to know the exact value of n. Finally, the normative distribution $\mathcal{D}$ is public, but the auditor does not know the alternative distribution $\mathcal{D}'$. The audited entity also knows $\mathcal{D}$ but may decide to train on $\mathcal{D}'$ for cost reasons. For instance, $\mathcal{D}$ could be the US population distribution and $\mathcal{D}'$ the distribution in the few locations that the entity selected for data collection.

4.2.3 Types of Audits

It now remains to discuss what the alternative distribution $\mathcal{D}'$ should be. Sometimes, the auditor wants to determine whether $\mathcal{D}'$ is different at all from the normative distribution $\mathcal{D}$. We call this the *distribution shift* (DS) setting, and here the audit is motivated by the concern that the use of an outdated training set may render a model ineffective.

Definition 4.3 (Distribution shift)**.** *The alternative distribution $\mathcal{D}'$ is shifted from the normative distribution $\mathcal{D}$ if $\mathcal{D} \neq \mathcal{D}'$.*

Another motivation for the audit is to detect a specific type of DS: a shift caused by underrepresentation, wherein a specific demographic group is not as prevalent in the model's training set as it should be. Definition 4.4 formalizes this *group distribution shift* (GDS) setting. Here, the random variable Z denotes the relevant demographic attribute, and X and Y are the input features and the response (or outcome), respectively.

Definition 4.4 (Group distribution shift)**.** *The alternative distribution $\mathcal{D}'$ is group-shifted from the normative distribution $\mathcal{D}$ if*

$$\Pr_{(x,y,z)\sim\mathcal{D}}[X{=}x, Y{=}y \mid Z{=}z] = \Pr_{(x,y,z)\sim\mathcal{D}'}[X{=}x, Y{=}y \mid Z{=}z]$$

for all possible values of x, y, and z, but the marginal distribution of Z is different for $\mathcal{D}$ and $\mathcal{D}'$, i.e., there exists z such that

$$\Pr_{(x,y,z)\sim\mathcal{D}}[Z{=}z] \neq \Pr_{(x,y,z)\sim\mathcal{D}'}[Z{=}z].$$

Throughout this chapter, we assume that Z is binary (i.e., $Z \in \{0, 1\}$) and that $Z = 1$ describes the underrepresented group.

GDS has a wide range of possible causes. For instance, it could be caused by a sampling bias. To illustrate this, consider the example of loan applications where the data collection for the training set overlooked poor neighborhoods. In the United States, poor neighborhoods disproportionately consist of racial minorities, so this data collection practice may lead to a racial sampling bias. Although this bias does not necessarily make the resulting model unfair, it may contribute to an inter-group disparity in accuracy where other types of DS would not, so we want the auditor to be able to detect GDS specifically.

4.3 The Auditing Technique

In this section, I describe and motivate the DS auditing procedure. The audit is essentially a controlled experiment that compares the audited model with *shadow models*, which are hypothetical models trained on $\mathcal{D}$. By training these shadow models, the auditor learns how a model would behave if its training set indeed follows the distribution $\mathcal{D}$. Then, the auditor trains an *attack model* to distinguish the input/output behavior of the audited model from those of the shadow models, and if the attack model is successful, we can infer that the audited model has a different training set distribution. The rest of this section elaborates on this setup and describes how to make this approach rigorous via hypothesis testing.

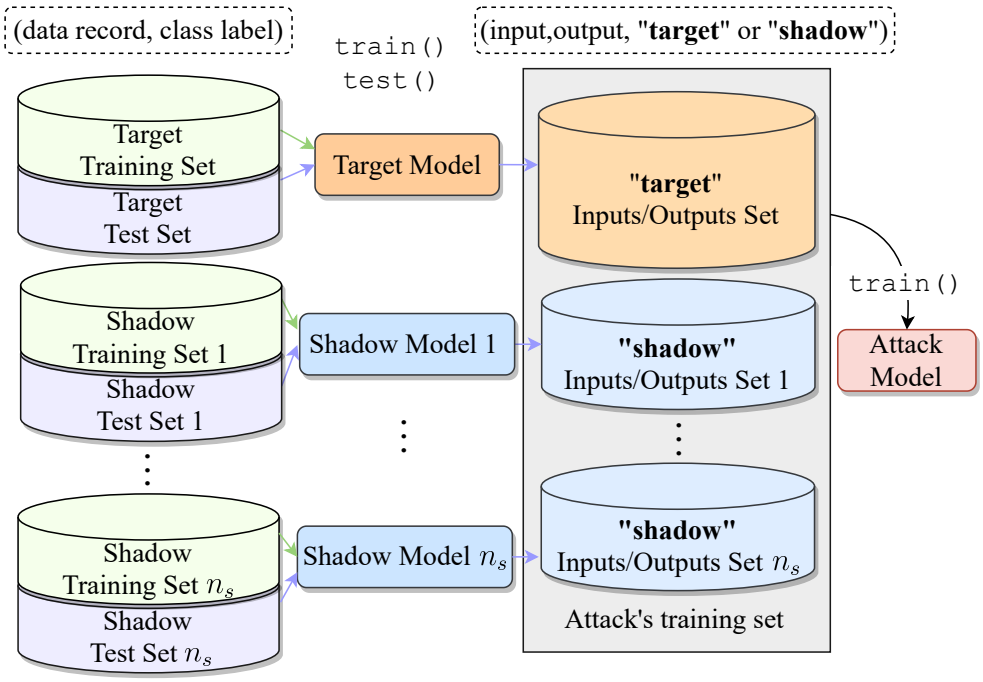


Figure 4.1: A schematic for training the attack model. To obtain the attack model's training set, we first train and test the target and shadow models with different datasets. Then, their inputs and outputs on a test set are labeled with whether the model was the target or a shadow model.

Section 4.3.1 provides a brief overview of the shadow model setup introduced by Shokri et al. [87] for membership inference. In Section 4.3.2, I explain how we adapt this setup for the DS audit. Finally, Section 4.3.3 provides a theoretical analysis that informs how the audit can be modified to perform a GDS audit.

4.3.1 The Shadow Model Attack

Shokri et al. [87] introduced a novel technique to procure examples for the membership inference attack described in Definition 4.1. To decide whether a challenge point (x, y) belongs to the training set, a membership inference attacker needs to learn from examples of cases where $b = 0$, in which points are drawn from the training set S of the target model h_S, and where $b = 1$, wherein they are drawn from the general population $\mathcal{D}$. However, the attacker cannot draw points from S because the target model's training set is unknown to him. Instead, he samples a new set S_i to train a *shadow model* h_{S_i} using the same learning algorithm A as the target model. Then, because the attacker knows the shadow model's training set, he can train an *attack model* that learns how the shadow model behaves differently on its training set compared to the general population.

This lets the attack model distinguish members of the shadow model training set from the general population. However, the attacker's ultimate goal is to identify the training set of the *target* model, not the shadow model. Therefore, the attacker has the attack model learn from multiple shadow models $h_{S_1}, \ldots, h_{S_{n_s}}$, all of which have different training sets. This makes it

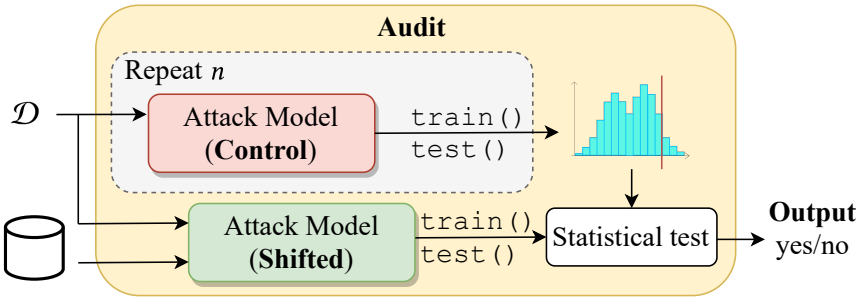


Figure 4.2: Diagram of the audit's workflow. The auditor has access to the public distribution $\mathcal{D}$ and draws samples to train the models in the control and shifted settings. The audited model (whose training is possibly drawn from $\mathcal{D}'$) is queried in the shifted setting only. The threshold is taken from the attack model's accuracy distribution in the control setting and applied in the statistical test to the attack model's accuracy in the shifted setting.

more likely that the attack model learns patterns that pertain to the learning algorithm A and generalize to the target model. Finally, at test time, the attacker gives the attack model the challenge point (x, y) as well as the model output $h_S(x)$, and the attack model guesses whether the challenge point is a part of the target model training set S.

4.3.2 Distribution Shift Audits

The main component of the DS audit is an attack model based on the shadow model attack described in the previous section. We now modify this attack for the DS audit game (Definition 4.2).

The attack model. The Shokri et al. [87] setup is not directly applicable to the DS audit game for two reasons. First, the auditor is no longer given a challenge point (x, y), so the attack model does not receive one either. Second, rather than learning to distinguish the members of the training set from the general population, the attack model learns to distinguish the target model from the shadow models. In particular, the attack model is given the input-output behavior of a model, which is equally likely to be the target model h_S or a shadow model h_{S_i}.

Fig. 4.1 depicts this setup—the attack model's training set consists of a set of input points and the corresponding outputs from the target and shadow models. Each input/output pair is then labeled with a bit indicating whether it came from the target model or a shadow model. Because a single input/output pair may not contain enough information, the attack model is trained with n_q pairs at a time, all generated from the same model. Finally, at test time, the attack model is given n_t pairs at a time and must guess whether they came from the target model or a shadow model. The number of queries at test time (n_t) is allowed to differ from the number of queries at training time (n_q).

Note that a trivial attack model can attain a 50% accuracy. Moreover, if $b = 0$, i.e., the training set S of the target model is drawn from $\mathcal{D}$, the behavior of the target model will be similar to those of the shadow models, so it will be difficult for an attack model to attain an accuracy significantly greater than 50%. Therefore, if the attack model wins significantly more than half of the time, the auditor may conclude that S was drawn from $\mathcal{D}'$. However, even when

$b = 0$, the random draws of the training set from $\mathcal{D}$ may make the target model significantly different from the shadow models, especially if the models tend to overfit to their training sets. In that case, the attack model may have an accuracy significantly greater than 50%, resulting in a false positive finding that S was drawn from $\mathcal{D}'$.

The controlled experiment. To address this issue, we take a hypothesis testing approach with the following null hypothesis:

H_0: The audited model's training distribution is $\mathcal{D}$.

To test H_0, the auditor performs a controlled experiment, which is divided into two settings:

- The *control* setting: Following the attack model setup described above, the auditor draws data from $\mathcal{D}$ to train a model, which we call the target model, and trains n_s shadow models, also with data drawn from $\mathcal{D}$.
- The *shifted* setting: The auditor follows the same steps as in the control setting, but she does not train the target model—the audited model becomes the target model.

In both settings, the auditor uses the inputs and outputs of the target and shadow models to train and test the *attack* model, as represented in Fig. 4.2. The control setting evaluates the attack model's performance when the target and shadow models' training sets are drawn from the same distribution. By repeating the experiment in the control setting multiple times, the auditor can measure the extent to which the attack model's performance can be attributed to the randomness in training (e.g., the specific training sample). In the shifted setting, the auditor is given only one model to audit (h_S in Definition 4.2), so she can only make one measurement of the attack model's performance. As in the original membership inference attack, the auditor can train multiple shadow models in both control and shifted settings to reduce the noise in her estimate of the attack model's performance.

The statistical test. To determine whether the difference in attack accuracy between the control and the shifted settings is significant, the auditor performs the following statistical test. First, she takes note of the distribution of the attack model's performance in the control setting and determines the threshold at which she would reject the null hypothesis. In Section 4.4.1, I describe how we choose the threshold for the empirical evaluation, but this is not intended to be prescriptive—the choice of the threshold should depend on the specific use case and the relative cost of false positives versus false negatives.

After setting the threshold, the auditor deploys the attack in the shifted setting, measures its accuracy, and applies the threshold. If the measured accuracy is above the threshold, the auditor can be reasonably confident that the training set S was not drawn from distribution $\mathcal{D}$.

4.3.3 Group Distribution Shift Audits

The approach described in the previous section allows us to detect DS in general, but it cannot determine whether the DS is a case of GDS. Thus, we modify the audit to detect GDS as defined in Definition 4.4. The new audit proceeds identically to that described in Section 4.3.2, but

instead of comparing the *performance* of the attack model to that in the control setting, the auditor compares the *inter-group difference* in performance.

I will now theoretically analyze such an auditor in a setting similar to the one I analyzed for membership inference [107]. This analysis is a heuristic argument that, at least for learning algorithms that tend to overfit to their training data, the inter-group difference in the attack model's performance is better suited for detecting an unrepresentative training set than the attack model's performance itself.

Consider a 1-dimensional input $X \in \mathbb{R}$, binary response $Y \in \{0, 1\}$, and binary protected attribute $Z \in \{0, 1\}$. For the $Z = 0$ group, the data distribution $\mathcal{D}_0$ is as follows:

- $\Pr[Y = 0] = \Pr[Y = 1] = \frac{1}{2}$
- $(X \mid Y{=}0, Z{=}0) \sim \mathcal{N}(-1, 1)$
- $(X \mid Y{=}1, Z{=}0) \sim \mathcal{N}(1, 1)$.

For the $Z = 1$ group, the data distribution $\mathcal{D}_1$ is

- $\Pr[Y = 0] = \Pr[Y = 1] = \frac{1}{2}$
- $(X \mid Y{=}0, Z{=}1) \sim \mathcal{N}(-1 + \tau, 1)$
- $(X \mid Y{=}1, Z{=}1) \sim \mathcal{N}(1 + \tau, 1)$.

Here, the parameter τ indicates the degree to which the two protected groups differ. Finally, the population data distribution $\mathcal{D}$ is a 50-50 mix of $\mathcal{D}_0$ and $\mathcal{D}_1$.

We now train the target model h_t and shadow model h_s. To simulate an unrepresentative training set, we will assume that all points in the target model training set S_t are drawn from $\mathcal{D}_0$, whereas the shadow model is trained from a balanced training set S_s that is drawn from $\mathcal{D}$. The attack model is given either the target model or the shadow model, each with probability 1/2, and it queries the model on a random point x to determine which model it was given.

In my analysis of membership inference [107], I assumed that the models tend to overfit to their training sets, having a mean squared regression error of σ^2_{train} when the input is in the training set S and a larger error σ^2_{test} otherwise. Here, because we have a classification setting, we instead assume that the model is correct with probabilities π_{train} and π_{test}, respectively, where $\pi_{\text{train}} > \pi_{\text{test}}$. Moreover, the probability that the randomly drawn query point x is in the training set is zero, so instead of requiring x to be in the training set to have the increased accuracy π_{train}, we only require that it be close to one of the points in the training set. Thus, we have

$$\Pr[h(x) = y] = \begin{cases} \pi_{\text{train}}, & \text{if } \exists x' \in S \text{ s.t. } \|x - x'\| \leq \epsilon \\ \pi_{\text{test}}, & \text{otherwise.} \end{cases}$$

Now, suppose that our theoretical attack model only uses information about the accuracy of the queried model. In particular, without loss of generality, the attack model guesses "target model" if the model h is correct on the queried point x, and "shadow model" otherwise. For brevity, let $f_t(\mathcal{Q}) = \Pr_{x \sim \mathcal{Q}}[\exists x' \in S_t \text{ s.t. } \|x - x'\| \leq \epsilon]$ and we define $f_s(\mathcal{Q})$ similarly, except

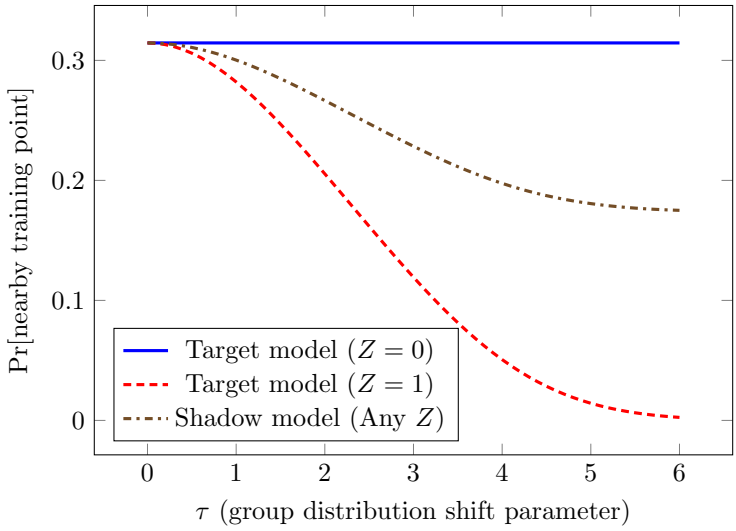


Figure 4.3: Probability that a random data point is close to a training set point for the models analyzed in Section 4.3.3. The target model has an unrepresentative training set, and the shadow model has a demographically balanced training set. If the models overfit to their training sets ($\pi_{\text{train}} > \pi_{\text{test}}$), this difference in probability can allow an auditor to detect a group distribution shift.

with S_s instead of S_t. Then, when using a query point $x \sim \mathcal{Q}$, the attack model's accuracy is

$$\begin{aligned}
&\tfrac{1}{2} \cdot \big(\Pr[\text{attack model guesses } h_t \mid \text{it was given } h_t] \\
&\qquad + \Pr[\text{attack model guesses } h_s \mid \text{it was given } h_s] \big) \\
=&\tfrac{1}{2} \cdot \big(\Pr_{x\sim\mathcal{Q}}[h_t(x) = y] + \Pr_{x\sim\mathcal{Q}}[h_s(x) \neq y] \big) \\
=&\tfrac{1}{2} \cdot \big(\pi_{\text{train}} f_t(\mathcal{Q}) + \pi_{\text{test}}(1 - f_t(\mathcal{Q})) \\
&\qquad + (1 - \pi_{\text{train}}) f_s(\mathcal{Q}) + (1 - \pi_{\text{test}})(1 - f_s(\mathcal{Q})) \big) \\
=&\tfrac{1}{2} + \tfrac{1}{2}(\pi_{\text{train}} - \pi_{\text{test}})(f_t(\mathcal{Q}) - f_s(\mathcal{Q})).
\end{aligned}$$

This means that the difference between the attack model's performance and random guessing is proportional to $f_t(\mathcal{Q}) - f_s(\mathcal{Q})$.

Fig. 4.3 plots the values of $f_t(\mathcal{Q})$ and $f_s(\mathcal{Q})$ for the case where $\epsilon = 0.001$ and the training set consists of 1,000 i.i.d. points drawn from the distributions described in the beginning of this section. Although these specific values are chosen for concreteness, the patterns shown here generalize to many different values of ϵ and training set sizes. Then, the solid blue line and the dashed red line correspond to $f_t(\mathcal{D}_0)$ and $f_t(\mathcal{D}_1)$, respectively. In addition, the brown dash-dotted line represents $f_s(\mathcal{D}) = f_s(\mathcal{D}_0) = f_s(\mathcal{D}_1)$.

Since the data distribution $\mathcal{D}$ is a 50-50 mix of $\mathcal{D}_0$ and $\mathcal{D}_1$, $f_t(\mathcal{D})$ is simply the average of the blue (solid) and the red (dashed) lines. This average is close to the brown (dash-dotted) line, so the quantity $f_t(\mathcal{D}) - f_s(\mathcal{D})$ is close to zero, which means that the attack model is not much more accurate than random guessing. However, if we consider the two protected groups separately, for sufficiently large GDS parameter τ, the quantity $f_t(\mathcal{D}_0) - f_s(\mathcal{D}_0)$ is a large positive number, and

$f_t(\mathcal{D}_1) - f_s(\mathcal{D}_1)$ is a large negative number. This means that, if the learning algorithm overfits, i.e., $\pi_{\text{train}} - \pi_{\text{test}}$ is sufficiently large, the inter-group difference in the attack model's performance will also be large.

The above theoretical analysis motivates the GDS audit in this chapter. The shadow model and attack model setup in both the control and shifted settings proceeds identically to that described in Section 4.3.2. However, instead of measuring the attack model's performance, the auditor measures how much more accurate the attack model is on points drawn from $\mathcal{D}_0$ compared to $\mathcal{D}_1$. She then sets a threshold based on this inter-group gap in the control setting and uses the threshold to decide whether GDS exists in the shifted setting. The experimental results in Section 4.5.2 show that this approach can indeed detect GDS.

4.4 Methodology

In the following experiments, we introduce a shift between $\mathcal{D}$ and $\mathcal{D}'$ to simulate the audit and measure its performance. In this section, I detail procedures that are common to all experiments, and all other details are given in the section where they are relevant.

4.4.1 Measurements

For both the control and shifted settings, we partition the data into five subsets—target model training set, shadow model training set, target/shadow model test set, attack model training set, and attack model test set. In the control setting, all partitions are drawn from $\mathcal{D}$, whereas in the shifted setting, the target model training set differs in that it is drawn from $\mathcal{D}'$. Although the training set of the audited model in Definition 4.3 can be drawn from either $\mathcal{D}$ or $\mathcal{D}'$ depending on the value of b, we always draw from $\mathcal{D}'$ in the shifted setting because the audit's performance when $b = 0$ can be measured through the control setting.

We run each setting multiple times with different random seeds, and the partitions change depending on the random seed. The exact number of random seeds is indicated in Section 4.4.3. Note that the auditor would only be given one model to audit during the deployment of the audit. However, to estimate the test's statistical power (true positive rate), we also repeat the shifted setting experiment multiple times. The statistical power indicates how likely the auditor is to correctly reject H_0.

We calculate the true positive rate by setting the auditor's threshold at the 90th percentile of the control performance, which corresponds to a 10% false positive rate. We apply this non-parametric approach, rather than a t-test as done by Maini et al. [71], because here we cannot assume that the control performance follows any specific distribution. To measure how effective the audit is with other thresholds, I also report the area under the receiver operating characteristic (ROC) curve.

4.4.2 Attack Models

Here, I describe the two types of attack model that were implemented for the empirical evaluation. We call these types *Complex* and *Simple*, referring to their relative complexity.

Complex attack model. The Complex attack model receives the input x to the target or shadow model h, as well as the output $h(x)$. More specific details about the model architecture are given in Section 4.4.3. When the number of queries n_q or n_t is greater than 1, the attack model first evaluates each query individually, arriving at a real-valued score for each query. Finally, the sum of these scores is sent through a sigmoid activation layer to reach a single probabilistic prediction. To evaluate the accuracy of this model, we use a threshold of 0.5 to binarize this probability.

In general, increasing the number of queries will make the attack model more accurate but take more time to train. To build an accurate attack model efficiently, we sometimes set the parameters so that the attack model makes more queries at test time than at training time, i.e., $n_t > n_q$.

Simple attack model. The Simple attack model is motivated by the theoretical analysis in Section 4.3.3. In that analysis, the attack model uses only the accuracy of the target or shadow model to determine which model it queried. Therefore, the Simple attack model is only given the average performance (accuracy for classification; mean squared error for regression) of the target or shadow model on the n_q or n_t queries it makes. The attack model then sets a threshold on the average performance to decide which model it queried.

To allow a direct comparison of the accuracy or mean squared error between training and testing, we always let $n_q = n_t$ for the Simple attack model.

4.4.3 Datasets and Model Architecture

In this section, I describe the datasets and model architecture that are used for the empirical evaluation.

Face datasets. We use two face datasets that are commonly used for training face recognition tasks: `UTKFace` [116] and `CelebA` [67]. We build a convolutional neural network (CNN) to classify faces by gender. In particular, the inputs to the target and shadow models are first sent through 1–2 convolutional layers with 3×3 kernels and then a max pooling layer. We continue alternating between 1–2 convolutional layers and a max pooling layer until the input has been sent through four max pooling layers. We then add a fully connected layer of 128 neurons, followed by a fully connected layer of 1 neuron with sigmoid activation. Except the final fully connected layer, all layers use ReLU activation.

The architecture of the Complex attack model is similar, but the output of the target or shadow model is concatenated to the output of the fourth max pooling layer and then together sent through the fully connected layers. We concatenate to this intermediate result rather than at the beginning because the output of the target or shadow model is a binary value that would not benefit from going through the convolutional and max pooling layers, which are designed for handling images. When the number of queries n_q or n_t is more than one, we first evaluate each query independently of one another up to, but not including, the sigmoid activation. Then, the sum of these results is converted to a probabilistic prediction via the sigmoid activation function.

The target and shadow models were trained for 20 epochs, and the attack models for 50 epochs, both with a batch size of 32. All experiments were run with 50 different random seeds

using Python 3.8.3 and TensorFlow 2.3.1 [1] on a Titan RTX GPU.

Medical dataset. The `Warfarin` [50] dataset contains medical, genetic, and demographic information about patients who were prescribed warfarin, as well as their warfarin dosages. The target and shadow models predict the correct warfarin dosage. The models consist of a fully connected layer of 32 neurons with ReLU activation, followed by a fully connected layer of 1 neuron. Because this is a regression task, we do not use any activation function after the final layer.

For the Complex attack model, the inputs and output of the target or shadow model are concatenated to become the input to the attack model. Unlike the target and shadow models, the attack model has a classification task, with "target" and "shadow" as the two class labels. Therefore, the output of the final layer is converted to a probability using the sigmoid activation function. When the number of queries n_q or n_t is more than one, the logit values for each query are added together before being sent through the sigmoid function.

The target and shadow models were trained for 100 epochs, and the attack models for 50 epochs, both with a batch size of 32. All experiments were run with 50 different random seeds using Python 3.8.3 and TensorFlow 2.3.1 [1] on a Titan RTX or Titan X (Pascal) GPU.

Internet security datasets. We consider two different use cases in the security domain: intrusion and malware detection. For malware detection, we use two different datasets: `Drebin` [8] and `Marvin` [64]. The DS between these two distributions has been previously studied in the malware literature [10, 47, 51]. For intrusion detection, we use the NSL-KDD dataset [96], a more realistic version of the original KDD'99 Cup dataset [26]. This dataset consists of a set of network events that are labeled with whether or not the event is an intrusion. There are four types of events but, because of the lack of data for the other events, we focus on Denial-of-Service (DoS) and Probing (Probe) events. The KDD'99 Cup challenge was to design a model that is robust to DS, so its training and test sets come from two different distributions. We name these distributions `DoS1`, `DoS2`, `Probe1`, and `Probe2`.

Unless otherwise stated, the target and shadow models are multi-layer perceptrons (MLPs). They have two hidden layers of 32 and 2 neurons with ReLU activation. The final, output layer is a fully connected layer of 1 neuron with sigmoid activation. Both target and shadow models were trained for 200 epochs with a batch size of 200.

For the intrusion detection models we use the features that prior work has found to be the most relevant [74]. For the malware models we also use features that are common in the malware literature [8, 51]. The Complex attack model for these experiments is identical to the Complex attack used in the warfarin dosing task. The experiments were run with 100 different random seeds, except the effect size experiments, which were run with 50 different seeds.

4.5 Evaluation

I now describe the experiments in detail and present their results. Section 4.5.1 deals with distribution shifts in general, whereas the audits in Section 4.5.2 are intended to detect *group* distribu-

tion shifts specifically. For both types of audits, I first highlight their use cases and then explore how the various aspects of the model and data affect the auditor's performance.

4.5.1 Distribution Shift Audits

DS audits are useful in situations where the auditor's goal is to verify that the audited model's training set comes from a fixed, public distribution $\mathcal{D}$. These audits test a more general hypothesis than the GDS audits (see Section 4.5.2), as they test for any type of shift between the audited model's training distribution and $\mathcal{D}$.

All the experiments in this section use the Complex attack model with $n_q = 10$ and $n_s = 1$. The results show that the audits achieve high statistical power in both use cases that we consider, even for moderate-to-small effect sizes. When we vary the target model's learning algorithm, we observe differences in the audit's performance; I identify several algorithmic properties that may be involved. We also evaluate the Simple attack on some of the data and observe similar audit performance. We consider the following two use cases:

Internet security applications. We consider the same type of audit for both malware and intrusion detection. The audits are performed on the malware or intrusion detection system, and the auditor tests whether the target model is up to date with the latest distribution $\mathcal{D}$ of malware binaries or network events.

Gender recognition. We explore an application of the DS audit for fairness. We consider a scenario in which a regulatory agency has standardized a normative distribution $\mathcal{D}$ for face recognition. Thus, the audited entity must comply and train their models with data from $\mathcal{D}$. The auditor applies the DS audit to check whether the audited entity is compliant.

Statistical Power

In these experiments, we ensure that $\mathcal{D}$ and $\mathcal{D}'$ are two different distributions. By taking distributions that have such an evident shift, we introduce a large effect in the shifted setting. Thus, we would expect the audit to achieve high true positive rates (TPR) in detecting these shifts.

Table 4.1 lists the setup of these experiments. On each run, we split the data into partitions of the same size to train and test the target, shadow and attack models. The train-test split is 50-50 for the gender recognition models and 60-30 for the malware and intrusion detection models. The splits are stratified to preserve the label distribution. This is especially important for the security applications, as they tend to be skewed towards the negative class. In addition, the DS may involve a shift in the label distributions, and the stratification ensures that the shift is preserved.

Table 4.1 also shows the performance of the target model on the test sets. I report the mean accuracy for the gender recognition task but, because of the imbalance in the classes, the F1-score is reported as an aggregate metric of performance for the malware and intrusion detection models. After training the target model on $\mathcal{D}'$ and testing it on $\mathcal{D}$, we observe a substantial drop in performance for all the tasks, which indicates a large shift between the distributions.

For the gender recognition task, we use data augmentation to generate the training images. Data augmentation transformations generate slightly different images on every training epoch,

Table 4.1: Experiments in Section 4.3.2 and their configurations. All experiments use the Complex attack with $n_s = 1$ and $n_q = 10$. The split size indicates the size of the data partitions into target, shadow, and attack sets. In all experiments, the target and shadow models are trained with the same learning algorithm (A). Finally, the last column is the performance of the target model in the control setting: I report the mean F1-score for intrusion and malware detection and mean accuracy for gender recognition.

Task	$\mathcal{D}$	$\mathcal{D}'$	**Split Size**	A	**Target Model Performance**
Gender recog.	`CelebA`	`UTKFace`	∼8K	CNN	93.50±0.83%
Malware detection	`Drebin`	`Marvin`	3K	MLP	97.71±1.04%
ID (DoS)	`DoS1`	`DoS2`	∼16.5K	MLP	98.25±3.96%
ID (Probe)	`Probe1`	`Probe2`	∼11.5K	MLP	96.89± 1.77%

homogenizing the datasets and eliminating visual differences stemming from image processing. We thereby discourage the attack model from picking up on visual cues to distinguish the datasets. We center-crop the CelebA images to 60% of their original size and resize the images in both datasets to the same size. This ensures that the images from both datasets are cropped similarly. In addition, we flip the image horizontally uniformly at random.

Note that, in practice, the auditor knows the image size that the target model expects because she has query access to it. Moreover, she can always resize and crop the images from the distribution $\mathcal{D}$.

In Table 4.2, I show the attack model's mean accuracy in the control and shifted settings. As expected, in the control setting, the attack model's guesses are close to random, except for the ID (Probe) task, which achieves 60% mean accuracy with slightly higher variance than the other tasks. However, even for the ID (Probe) task, the attack model's mean accuracy in the shifted setting is notably higher.

Fig. 4.4 contains the histograms of the attack model accuracy in the control and shifted settings for the ID (Probe) task. The small overlap between the histograms indicates high statistical power for the auditor. Although the attack model sometimes exhibits >90% accuracy in the control setting, this is rare enough that the audit achieves >90% TPR on all tasks with a 10% false positive rate (FPR). In other words, the auditor correctly concludes that the target model was not trained on $\mathcal{D}$ more than 90% of the time. Table 4.2 also shows the area under the ROC curve (AUC). The high AUC values imply that the auditor can reduce the FPR even further without an excessive decrease in TPR.

Effect Size

The shift between $\mathcal{D}$ and $\mathcal{D}'$ in the previous section is very large, so the high TPRs are not too surprising. We now evaluate the audit for smaller shifts. To simulate a gradually increasing shift,

Table 4.2: Summary of the DS audit results. I report the attack model's mean accuracy (and standard deviation) in the control and shifted settings, followed by the AUC, and the TPR in the 90th percentile of the control scores.

	Accuracy			
Task	**Control**	**Shifted**	**AUC**	**TPR**
Gender recognition	51.51±3.82%	98.44±1.10%	100%	100%
Malware detection	50.39±5.62%	97.16±6.53%	99.35%	98%
ID (DoS)	51.98±7.70%	79.35±4.99%	98.00%	100%
ID (Probe)	60.74±12.26%	85.68±5.63%	96.27%	94%

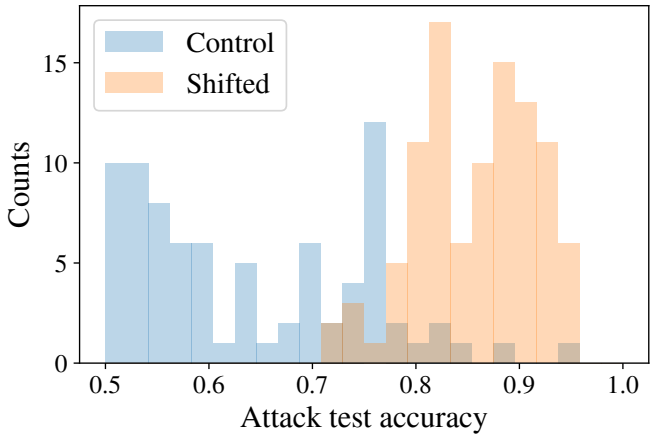


Figure 4.4: Histograms of the control and shifted mean attack model accuracy for the ID (Probe) task.

Table 4.3: The attack model's mean accuracy (and standard deviation) in the control and shifted settings for a convex combination of `DoS1` and `DoS2`. The audit's AUC and TPR in the 90th percentile threshold follow. α parametrizes the mixture distribution $\mathcal{D}$.

	Accuracy			
$\boldsymbol{\alpha}$	**Control**	**Shifted**	**AUC**	**TPR**
0.00	53.05±5.74%	77.07±7.27%	99.00%	96%
0.28	52.71±3.85%	70.08±5.05%	99.64%	100%
0.52	52.31±4.10%	65.68±7.32%	94.56%	86%
0.76	50.98±4.56%	57.79±6.06%	87.00%	70%
1.00	52.86±8.65%	54.01±10.19%	55.96%	6%

we take $\mathcal{D}$ to be a convex combination of the two original distributions, which we denote by $\mathcal{D}_*$ and $\mathcal{D}'_*$:

$$\mathcal{D} = \alpha\,\mathcal{D}_* + (1-\alpha)\,\mathcal{D}'_* \quad \text{and} \quad \mathcal{D}' = \mathcal{D}_*.$$

Here, α can take any value in $[0, 1]$. To clarify, we are not taking the weighted sum of pairs of instances, but rather sampling from the original distributions in proportion to α.

Table 4.3 shows the attack model's accuracy for the ID (DoS) task. The split size is 4,293 points. Compared to the previous experiments, the split size of 4,293 is smaller because we need to combine data from both `DoS1` and `DoS2` to construct the sample from $\mathcal{D}$. We reserve 4,293 instances from `DoS1` to train the target model in the shifted setting and the rest are mixed with `DoS2`, from which we draw the sets for the other models. For all the models, we divide each of the sets 50-50 into training and testing. The different split size and train-test ratio between this experiment and the previous ones may account for the slight decrease in the attack model's accuracy at $\alpha = 0$.

The attack model's accuracy in the control setting is around 0.5 for all α. We observe that the accuracy in the shifted setting is approximately linear in α, with a Pearson correlation coefficient of -0.99 for 30 equidistant points of α. However, the auditor's performance is not linear over α. For an effect approximately half the size of the original effect ($\alpha = 0.52$), the AUC is 94.56%, and for $\alpha = 0.76$, the AUC is 87.00%. This shows that the audits achieve high statistical power even for moderate-to-small effect sizes, despite the decrease in attack accuracy. Moreover, the linearity of the shifted accuracy with α implies that the auditor can estimate the magnitude of the shift simply by measuring the shifted accuracy.

Learning Algorithm

The learning algorithm used to train the target model has an impact on the audit's performance. If the learning algorithm yields different models depending on the samples used for training, it can boost the attack model's accuracy. Conversely, a learning algorithm that yields similar models for samples coming from different distributions will be less vulnerable to the attack in the shifted setting. We now test different learning algorithms and identify the relevant algorithmic properties that play a role in the audit.

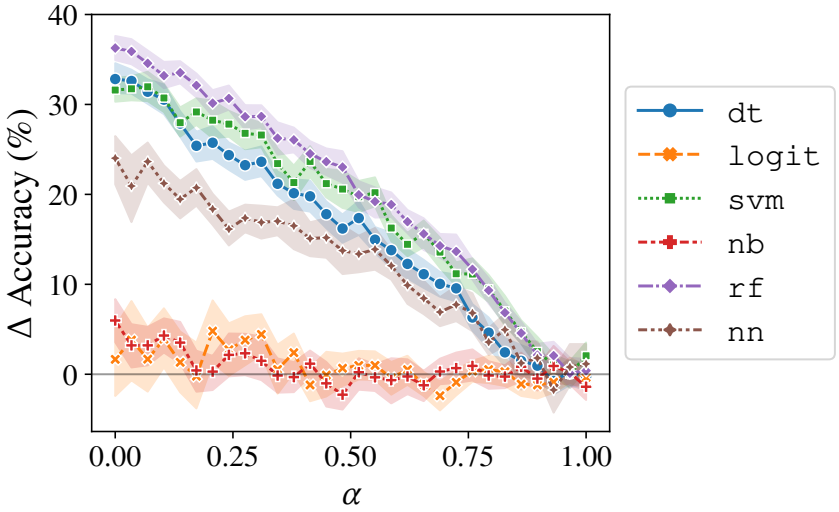


Figure 4.5: ΔAccuracy is the mean difference in attack accuracy between the control and shifted settings in the ID (DoS) experiment. α parametrizes the mixture distribution $\mathcal{D}$. The error bands are the 95% confidence intervals of ΔAccuracy.

The setup of the experiments here is exactly the same as in the effect size experiments presented earlier, except that we vary the learning algorithm. The learning algorithms that we test are: a CART Decision Tree (`dt`) with a maximum depth of five levels; a Logistic Regressor with L^2 regularization and a limited-memory BFGS solver (`logit`); a Support Vector Machine with an RBF kernel (`svm`); a Naive Bayes (`nb`) with different event models depending on the dataset (Gaussian for intrusion detection and Bernoulli for malware); a Random Forest with 50 estimators (`rf`); and a Multi-Layer Perceptron with the Adam optimizer (`nn`). We use the algorithms implemented by the sklearn library [76] in Python, and unless otherwise stated, we use their default parameters.

The results of the ID (DoS) experiment are shown in Table 4.4, and Fig. 4.5 plots the same for various values of α. The attack model's accuracy in the control setting is around 0.5 for all α. Thus, we again observe from the figure that the accuracy in the shifted setting is approximately linear in α for all learning algorithms. We also see that the accuracy depends heavily on the learning algorithms as well. In particular, the accuracy is substantially lower for `logit` and `nb` than the other algorithms. Yet, Table 4.4 shows that the audit's test has high statistical power even for these two algorithms. This indicates that the attack model's performance is sufficient for a successful DS audit. However, this is not the case for higher values of α. At $\alpha = 0.87$, the auditor is reduced to random guessing with these two algorithms, but she still achieves an AUC greater than 80% with the others.

Learning capacity. The lower performance of the audit when the target model is trained with `logit` and `nb` can be explained by the relatively low capacity of these algorithms. When we train the target model on `DoS2` and test on `DoS1`, `nb` and `logit` models have some of the

Table 4.4: Effect of various target model learning algorithms on the audit. I report the target model's performance (and standard deviation) in the control setting, as well as the attack model's mean accuracy (and standard deviation) in the control and shifted settings. This is followed by the AUC and the TPR in the 90th percentile of the control scores.

		Attack Accuracy			
Task	***A***	**Control**	**Shifted**	**AUC**	**TPR**
Malware detection	dt	49.81±5.77%	97.64±1.67%	100%	100%
	logit	49.70±5.55%	98.20±1.41%	100%	100%
	nb	49.68±5.83%	99.97±0.25%	100%	100%
	nn	50.39±5.62%	97.16±6.53%	99.35%	98%
	rf	49.73±5.62%	97.65±1.66%	100%	100%
	svm	49.39±5.59%	98.21±1.39%	100%	100%
ID (DoS)	dt	50.49±3.69%	84.59±4.22%	100%	100%
	logit	49.88±3.54%	66.34±6.75%	97.97%	92%
	nb	50.24±4.13%	63.94±6.16%	96.67%	95%
	nn	51.98±7.70%	79.35±4.99%	98.00%	100%
	rf	49.83±3.39%	86.86±3.30%	100%	100%
	svm	49.87±3.74%	84.45±4.73%	100%	100%
ID (Probe)	dt	50.06±4.19%	99.47±0.67%	100%	100%
	logit	50.68±4.42%	91.91±2.32%	100%	100%
	nb	65.21±14.18%	79.11±5.14%	77.75%	16%
	nn	60.74±12.26%	85.68±5.63%	96.27%	94%
	rf	50.24±4.25%	83.96±4.24%	100%	100%
	svm	50.31±4.64%	89.88±2.96%	100%	100%

Table 4.5: Generalization error of the target model for different learning algorithms in the ID (DoS) convex combination experiment for $\alpha = 0$.

	Target F1-Score	
A	**Shifted Train**	**Shifted Test**
dt	99.58±0.21%	80.32±4.76%
logit	98.03±0.26%	83.04±0.92%
nb	97.33±0.37%	83.32±0.61%
nn	96.38±11.92%	79.48±12.69%
rf	100±0.01%	79.61±2.23%
svm	98.46±0.22%	74.38±5.44%

lowest shifted train F1-scores but also the highest shifted test F1-scores (see Table 4.5). This suggests that these are simple models that underfit the training data. Because there is not as much difference between the models' training and test performance, the attack model is not as accurate on `nb` and `logit`.

Variance. I also highlight the `nn` rows in Table 4.4, which show that the `nn` algorithm tends to result in high mean and standard deviation of the attack model's accuracy in the control setting. In addition, Table 4.5 shows that the standard deviation in the target model's F1-score is one order of magnitude higher for `nn` in the ID (DoS) experiment. A closer look at individual F1-scores reveals that the `nn` failed to converge for a few seeds. When this happens, the attack model can easily distinguish between the target and shadow models, raising both the mean and standard deviation of the attack model's accuracy.

Data characteristics. Table 4.4 shows that the attack model's control accuracy for `nb` models in the ID (Probe) task is unusually high. When I looked at the F1-scores of these models, I found high variance. ID (Probe) has more imbalanced classes than the ID (DoS), and precision accounts for most of the variance in the F1-score. A closer look at the features in the positive examples reveals that some of these features have outliers. These outliers have high influence on the resulting model, as when they are present in the training set, the feature's mean and variance change significantly, drastically changing the target model. This in turn makes it easy for the attack model to distinguish the target model from shadow model even in the control setting. Thus, the interplay between the algorithm and the data also affects the audit.

4.5.2 Group Distribution Shift Audits

I now describe the GDS audit experiments. Recall from Definition 4.4 that GDS occurs when some demographic group is underrepresented in the alternative distribution $\mathcal{D}'$ compared to the normative distribution $\mathcal{D}$. Following the heuristic argument in Section 4.3.3, the auditor will use the inter-group difference in the performance of the attack model to decide whether GDS has occurred. Thus, if the attack model performs well on all people, this is not considered evidence of GDS, but if it performs significantly better on one demographic group, the auditor will suspect GDS even if the overall performance is not much better than random guessing.

I show that the Simple attack is successful in detecting GDS in both use cases that we consider, with decreasing statistical power for smaller effect sizes. In line with the theoretical anlysis in Section 4.3.3, the Simple attack is more successful than the Complex one in cases where the target model overfits. I also show that the GDS audit is significantly better than a naive audit that treats all instances of accuracy disparity as evidence of underrepresentation. Finally, I demonstrate that the GDS audit is unlikely to output a false positive in the presence of other types of DS. We consider two main use cases for this evaluation:

Gender recognition. It is often desirable that a face recognition model be equally accurate on all subgroups of the population. If this property does not hold, the auditor should ideally be able to point to a cause of the accuracy disparity so that specific remedial steps can be taken. Here,

Table 4.6: Experiments in Section 4.3.3 and their configurations. In all the experiments we use $n_s = 10$. The split size indicates the size of the data partitions into target, shadow, and attack sets. $\mathcal{D}_0$ and $\mathcal{D}_1$ are the overrepresented and underrepresented groups, respectively, and in all the experiments, the target and shadow models are trained with the same learning algorithm (A). For `UTKFace` and `CelebA`, n_q depends on whether the attack model is Simple (S) or Complex (C).

Task	**Dataset**	$\mathcal{D}_0$	$\mathcal{D}_1$	**Split Size**	n_q	n_t	A
Gender recog.	`UTKFace`	White	non-White	∼3.4K	100 (S) 10 (C)	100	CNN
Gender recog.	`CelebA`	young	old	∼18.4K	100 (S) 10 (C)	100	CNN
Medical dosing	`Warfarin`	White	Asian	∼600	10	10	MLP

the auditor specifically tests for an unrepresentative training set as a potential cause. Therefore, unlike in the setting described in Section 4.5.1, in this setting we consider a distribution shift to be an issue if and only if it changes the marginal distribution of the protected attribute (e.g., race or age).

Medical dosing. We consider dosing models that predict how much medicine should be prescribed to a given patient. For example, the International Warfarin Pharmacogenetics Consortium [50] published such an algorithm for the drug warfarin. Because the effectiveness of a drug may depend on the patient's race, it is important that the training sets of such models be racially diverse. We train and audit a hypothetical dosing model to evaluate whether the auditor can detect a racial imbalance in the model's training set.

Statistical Power

Table 4.6 lists the datasets and parameters used for training the target/shadow (learning algorithm A) and attack (number of queries n_q and n_t) models. Unless otherwise stated, the normative distribution $\mathcal{D}$ is a 50-50 mix of $\mathcal{D}_0$ and $\mathcal{D}_1$, and the alternative distribution $\mathcal{D}'$ simply equals $\mathcal{D}_0$. Thus, the goal of the auditor is to detect a target model whose training set underrepresents the $\mathcal{D}_1$ group.

The results of the experiments are given in the first three rows of Table 4.7. The table shows that, in the shifted setting, the target model may exhibit higher accuracy and lower mean squared error on $\mathcal{D}_0$ compared to $\mathcal{D}_1$, resulting in a disparate performance across groups. This is not the case in the control setting, and the training accuracy was significantly higher than the test accuracy, suggesting that the models tend to overfit to their training sets. This overfitting is consistent with the theoretical analysis in Section 4.3.3, and as a result the audit using the Simple attack model has much success. By contrast, the Complex attack model does not always lead to a successful audit. In particular, the AUC and TPR for the UTKFace audit using the Complex attack

Table 4.7: Results of the GDS audits. I report the target model's performance (accuracy for `UTKFace` and `CelebA`, mean squared error for `Warfarin`) and standard deviation for the two protected groups. This is followed by the audit's AUC and true positive rate for Simple (S) and Complex (C) attack models.

	Target Model Performance							
Experiment	**Control $\mathcal{D}_0$**	**Control $\mathcal{D}_1$**	**Shifted $\mathcal{D}_0$**	**Shifted $\mathcal{D}_1$**	**Atk**	**AUC**	**TPR**
UTKFace	83.05±4.27%	82.34±4.55%	85.04±1.56%	79.05±2.18%	S	79.68%	64%
					C	51.30%	12%
CelebA	93.95±0.69%	95.43±0.44%	95.36±0.45%	92.43±1.41%	S	99.96%	100%
					C	68.00%	40%
Warfarin 100%	1.35±0.32	0.92±0.10	1.15±0.22	6.77±1.49	S	100%	100%
					C	100%	100%
Warfarin 90%	1.35±0.32	0.92±0.10	1.19±0.22	1.52±0.28	S	91.74%	82%
					C	91.36%	82%
Warfarin 80%	1.35±0.32	0.92±0.10	1.21±0.22	1.12±0.16	S	62.86%	38%
					C	67.64%	42%
Warfarin 70%	1.35±0.32	0.92±0.10	1.23±0.22	1.00±0.14	S	57.80%	24%
					C	59.18%	24%
Warfarin 60%	1.35±0.32	0.92±0.10	1.27±0.22	0.95±0.12	S	49.40%	12%
					C	52.64%	14%
Other Shift	91.98±5.94%	93.66±2.76%	69.49±23.78%	72.39±23.79%	S	38.96%	8%
					C	33.46%	10%

is indistinguishable from the random guessing baselines of 50% and 10%, respectively, whereas the Simple attack leads to a significantly better outcome. I suspect that this difference arises because the Simple model is only given the accuracy of the target or shadow model, which is often sufficient to distinguish the target from the shadow. Then, because of the inter-group disparity in the target model's performance, the attack model's behavior on $\mathcal{D}_0$ and $\mathcal{D}_1$ are different, and the auditor can leverage this difference. On the other hand, the Complex model must learn the difference from scratch, a task at which it does not always succeed.

Effect Size

The above experiments involved target models that include *no* training examples from the underrepresented group. However, this assumption is not necessarily realistic because in practice we would expect *some* examples from a group even when it is underrepresented. To capture this use case, we now modify the alternative distribution $\mathcal{D}'$. As before, the normative distribution $\mathcal{D}$ is still a 50-50 mix of $\mathcal{D}_0$ and $\mathcal{D}_1$, but now only an α fraction of $\mathcal{D}'$ is $\mathcal{D}_0$, and the rest is distributed as $\mathcal{D}_1$. We follow the same experimental procedure as before with varying values of α for the medical dosing task to evaluate how the auditor's performance varies with α.

The fourth through seventh rows of Table 4.7 show that the audit becomes less successful as α approaches 0.5. This is to be expected because, when $\alpha = 0.5$, the target model in the "shifted" setting follows the normative distribution, leaving the auditor with no means to distinguish the control setting from the shifted setting. Indeed, when $\alpha = 0.6$, the auditor's performance is not significantly different from random guessing, but when $\alpha = 0.9$, she has a much higher likelihood of correctly detecting the GDS. Therefore, if the auditor audits a model and concludes that there is a GDS, this conclusion is likely to be indicative of a relatively large deviation from the normative distribution.

Comparison to a Naive Audit

We now compare the GDS audit to a naive audit that simply claims that the training set is unrepresentative if the accuracy disparity between groups is sufficiently large. To evaluate the naive audit, we measure the difference in the accuracy of target model for the $\mathcal{D}_0$ and $\mathcal{D}_1$ groups in both control and shifted settings. We then use these accuracy disparities to compute the AUC-ROC value of the naive audit.

The AUC values were 84.76%, 40.84%, 13.48%, and 5.00% for the Warfarin 90%, 80%, 70%, and 60% experiments, respectively. These results compare unfavorably with those reported in Table 4.7, indicating that this naive audit is insufficient for detecting group shifts except in extreme cases. In addition, for lower values of α, the sub-50% AUC shows that the naive audit is *more likely to be wrong than right*. This is because, as Table 4.7 shows, the target model tends to be more accurate on the $\mathcal{D}_1$ group than the $\mathcal{D}_0$ group. Moreover, if the auditor relies solely on accuracy disparity, she can be misled by other sources of accuracy disparity, such as the learning algorithm [46] and features [20].

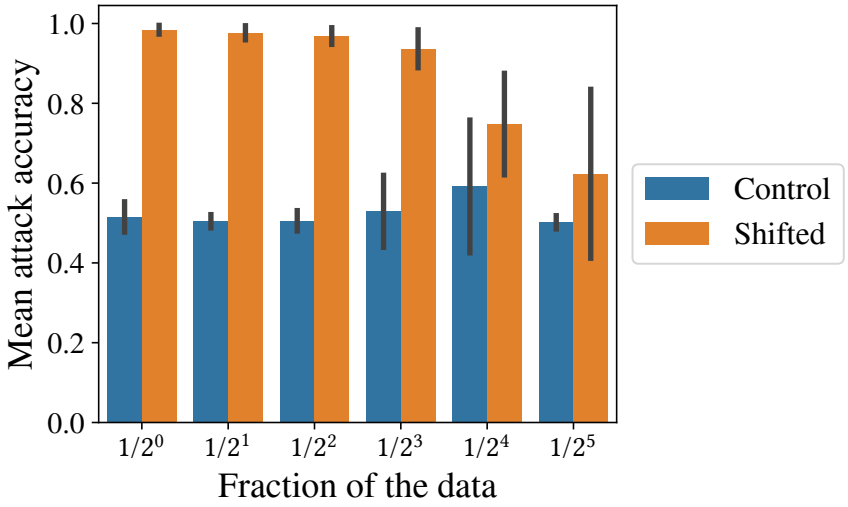


Figure 4.6: Mean attack accuracy in the control and shifted settings for different dataset sizes. The error bars indicate the standard deviation.

Presence of Other Shifts

Finally, we run an experiment to verify that the GDS auditor specifically detects *group* distribution shifts, i.e., that it does not detect other forms of DS. For this experiment, we induce a shift from $\mathcal{D}$ to $\mathcal{D}'$ without changing the marginal distribution of the protected attribute. In particular, $\mathcal{D}$ is a 50-50 mix of young and old faces, drawn from the CelebA dataset. $\mathcal{D}'$ is still a 50-50 mix of young and old faces, but these faces are drawn from the combined CelebA and UTKFace dataset. Because CelebA and UTKFace have different image sizes, all images were cropped to the same size for both $\mathcal{D}$ and $\mathcal{D}'$.

The final row of Table 4.7 contains the results of this experiment. The target model has an unusually high variance in the shifted setting—this is because the target model sometimes fails to learn patterns that apply to both UTKFace and CelebA. When this happens, the attack model attains near-perfect accuracy on both young and old faces. However, this results in no inter-group difference in the attack model's performance with respect to age, so we do not consider this an evidence of underrepresentation. As a result, the auditor's true positive rate is indistinguishable from random guessing, suggesting that this auditor specializes in detecting *group* distribution shifts.

4.5.3 Available Data

In Definition 4.2, we assume that the auditor knows the size of the target model's training set. Consequently, in all previous experiments we have ensured that the training sets of all models have similar sizes. Now we challenge this assumption and study the setting where the auditor has limited amount of data with which to train and test the models under her control. Less data may result in higher variance in the estimations of the attack model's accuracy, which in turn

may lead to a higher false positive rate for the audit. To measure whether this is indeed the case, we repeat the gender recognition experiment in Table 4.1 for power-of-two fractions of the initial dataset sampled from $\mathcal{D}$.

Fig. 4.6 shows that the gap in the mean attack accuracy between the control and shifted settings decreases with smaller fractions of the data. As expected, the variance in both control and biased accuracy increases with less data. Yet, even for a $1/8$ fraction of the original data, the difference between mean control and shifted accuracy is statistically significant, showing that the auditor can be successful with up to 8 times less data in the gender recognition task. Therefore, the assumption of knowing the exact size of the target model's training set can be relaxed in practice.

4.6 Conclusion and Discussion

The work in this chapter is the first to study black-box auditing techniques for DS in an adversarial setting. The evaluation in Section 4.5 shows that the proposed techniques are successful in a wide range of scenarios. In particular, GDS audits provide a tool for journalists and researchers to detect underrepresentation of minority groups in proprietary datasets. When applied in the wild, this tool can explain the accuracy disparities observed in commercial models. I now give some caveats and elaborate on some of the issues one may encounter when deploying the DS and GDS audits.

Audits in practice. In the deployment of these techniques, we must consider a model holder who will try to detect whether they are being audited and even deceive the auditor. I do not think such a model holder can reliably detect the audit. The number of queries required by the audit is not abnormally large, and these queries are drawn from the public normative distribution $\mathcal{D}$. Therefore, it is expected that other use cases will also involve querying the target model according to $\mathcal{D}$.

The results of the evaluation show that the audit has high recall (true positive rate), i.e., it is successful in detecting a shift given that there has been a shift. On the other hand, we do not measure the audit's precision, i.e., the likelihood of a shift given that the audit indicated that there is a shift. This quantity would depend not only on the audit's true and false positive rates, but also on the prior probability of a shift. It is out of the scope of this work to measure the prevalence of shifts in commercial models. However, in the settings where we obtained near-perfect AUC-ROC, the prior probability of a shift would need to be negligible for it to negatively affect the audit's precision.

Group distribution shift audits. The GDS audits may reveal a flawed data collection methodology that overlooks one of the groups and results in an accuracy disparity during deployment. I argue that detecting and acknowledging the cause of the disparity is the first step in any mitigation strategy. Audits that are not able to attribute the causes can be easily dismissed for not being sufficiently compelling, as they fail to show that the origin of the bias is the model holder's responsibility. For example, in the light of an accuracy disparity, the model holder could claim

that solving the task for one of the groups is inherently harder. A finding that GDS has occurred points to a specific action the model holder could take to mitigate the accuracy disparity.

Although the theoretical analysis in Section 4.3.3 assumes that models are more accurate near points in its training set, this assumption often holds in practice. For example, deep neural networks are capable of memorizing the entire training set and can fit random labels [112], so we would expect the model to be more accurate near training points.

Finally, underrepresentation is neither necessary nor sufficient for an accuracy disparity. The GDS audits do not verify a specific fairness definition in the model outcomes but rather detect an issue of underrepresentation. The Warfarin 80% experiment suggests that we can detect an underrepresentation issue even when there is no accuracy disparity. Even though such situations may appear to be harmless, the success of the audit indicates that target model's behavior is different across groups. This difference may translate to some form of unfairness even if there is no accuracy disparity. In addition, it is important to recognize the learning algorithm [46] and other parts of the machine learning pipeline [20] as potential sources of unfairness.

Chapter 5

Conclusion

This thesis attempts to reconcile two facts in fair machine learning: First, black-box techniques are especially useful because they are accessible to external entities. Second, there are many aspects of fairness that are not well captured by statistical parity. These facts are somewhat in conflict because, while statistical parity allows for black-box audits, it ignores some forms of blatant discrimination. Conversely, other techniques in fair machine learning can address these forms of discrimination, but they tend to be inaccessible to external entities. I have explored this issue for three forms of unfairness, presenting black-box techniques that enable external entities to gain a more comprehensive understanding of a model's behavior.

These techniques are important because of a misalignment in incentives. When a company trains a machine learning model to make decisions about humans, the model learns to maximize some objective chosen by the company, which often boils down to profit. If the model's decision unfairly harms an individual, that individual would be highly motivated to fix the harm, but the company could be indifferent, as it chose the objective that the model maximizes. Other considerations for the company include bad public relations and legal consequences, but both of these require the unfair behavior to be uncovered and demonstrated first. Therefore, black-box techniques fill a critical role of allowing external entities to analyze the model for unfair behavior.

As for statistical parity, it is a simple and useful notion of fairness, but it is neither necessary nor sufficient for avoiding discrimination. It is not necessary because a demographically unbalanced outcome can be justified if there is a sufficiently good reason, such as a "business necessity" [89]. For example, when hiring for a job that requires heavy physical labor, we would expect to hire more men than women, and this does not necessarily constitute discrimination. Moreover, statistical parity is not sufficient because there are many other aspects of unfairness, of which I have explored but three.

Although we cannot reasonably expect to eliminate all forms of unfairness, it is still important to mitigate some of them. Because many definitions of fairness cannot be satisfied simultaneously [22, 59], fair machine learning necessarily involves making trade-offs. Ultimately, discrimination is a societal issue, and addressing it will require the society as a whole to contemplate these trade-offs. My hope is that the technical tools presented here will spur honest, good-faith discussions on how we can leverage machine learning in a way that benefits everyone.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems. `https://www.tensorflow.org/`, 2015.

[2] Philip Adler, Casey Falk, Sorelle A Friedler, Tionney Nix, Gabriel Rybeck, Carlos Scheidegger, Brandon Smith, and Suresh Venkatasubramanian. Auditing black-box models for indirect influence. *Knowledge and Information Systems*, 54(1):95–122, 2018.

[3] Jason Altschuler, Jonathan Weed, and Philippe Rigollet. Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration. In *Neural Information Processing Systems*, pages 1964–1974, 2017.

[4] Martin S Andersen, Joachim Dahl, and Lieven Vandenberghe. CVXOPT: Python software for convex optimization. `http://cvxopt.org`.

[5] Ralph G Andrzejak, Klaus Lehnertz, Florian Mormann, Christoph Rieke, Peter David, and Christian E Elger. Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. *Physical Review E*, 64(6):061907, 2001.

[6] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. *ProPublica*, 2016.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.

[8] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, and Konrad Rieck. DREBIN: Effective and explainable detection of android malware in your pocket. In *Network and Distributed System Security Symposium*, 2014.

[9] Jeff Asher and Rob Arthur. Inside the algorithm that tries to predict gun violence in Chicago. *The New York Times*, 2017.

[10] Federico Barbero, Feargus Pendlebury, Fabio Pierazzi, and Lorenzo Cavallaro. Transcending transcend: Revisiting malware classification with conformal evaluation. *arXiv preprint arXiv:2010.03856*, 2020.

[11] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. `http://www.fairmlbook.org`.

[12] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[13] Emily Black, Samuel Yeom, and Matt Fredrikson. FlipTest: Fairness testing via optimal transport. In *ACM Conference on Fairness, Accountability, and Transparency*, pages 111–121, 2020.

[14] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[15] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on Fairness, Accountability and Transparency*, pages 77–91, 2018.

[16] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. In *Neural Information Processing Systems*, pages 3992–4001, 2017.

[17] Ramiro Camino, Christian Hammerschmidt, and Radu State. Generating multi-categorical samples with generative adversarial networks. *arXiv preprint arXiv:1807.01202*, 2018.

[18] Ran Canetti, Aloni Cohen, Nishanth Dikkala, Govind Ramnarayan, Sarah Scheffler, and Adam Smith. From soft classifiers to hard decisions: How fair can we be? In *ACM Conference on Fairness, Accountability, and Transparency*, pages 309–318, 2019.

[19] Gengxiang Chen, Yingguang Li, and Xu Liu. Transfer learning under conditional shift based on fuzzy residual. *IEEE Transactions on Cybernetics*, 2020.

[20] Irene Y Chen, Fredrik D Johansson, and David Sontag. Why is my classifier discriminatory? In *Neural Information Processing Systems*, pages 3543–3554, 2018.

[21] François Chollet et al. Keras. `https://keras.io`, 2015.

[22] Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data*, 5(2):153–163, 2017.

[23] Alexandra Chouldechova and Aaron Roth. The frontiers of fairness in machine learning. *arXiv preprint arXiv:1810.08810*, 2018.

[24] City of Chicago. Strategic Subject List. `https://data.cityofchicago.org/Public-Safety/Strategic-Subject-List/4aki-r3np`, 2017.

[25] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320, 2019.

[26] DARPA Intrusion Detection Evaluation Program, MIT Lincoln Laboratory. KDD cup 1999 data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 1999.

[27] Jeffrey Dastin. Amazon scraps secret AI recruiting tool that showed bias against women. *Reuters*, 2018.

[28] Amit Datta, Michael Carl Tschantz, and Anupam Datta. Automated experiments on ad privacy settings. *Privacy Enhancing Technologies*, 2015(1):92–112, 2015.

[29] Anupam Datta, Shayak Sen, and Yair Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *IEEE Symposium on Security and Privacy*, pages 598–617, 2016.

[30] Anupam Datta, Matt Fredrikson, Gihyuk Ko, Piotr Mardziel, and Shayak Sen. Proxy discrimination in data-driven systems. *arXiv preprint arXiv:1707.08120*, 2017.

[31] Anupam Datta, Matt Fredrikson, Gihyuk Ko, Piotr Mardziel, and Shayak Sen. Use privacy in data-driven systems: Theory and experiments with machine learnt programs. In *ACM SIGSAC Conference on Computer and Communications Security*, pages 1193–1210, 2017.

[32] Eustasio del Barrio, Fabrice Gamboa, Paula Gordaliza, and Jean-Michel Loubes. Obtaining fairness using optimal transport theory. *arXiv preprint arXiv:1806.03195*, 2018.

[33] Dheeru Dua and Efi Karra Taniskidou. UCI machine learning repository. https://archive.ics.uci.edu/ml, 2017.

[34] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Innovations in Theoretical Computer Science*, pages 214–226, 2012.

[35] Equal Employment Opportunities Commission. Uniform guidelines on employee selection procedures. 29 CFR Part 1607, 1978.

[36] Equivant. Practitioner's guide to COMPAS core. http://www.equivant.com/wp-content/uploads/Practitioners-Guide-to-COMPAS-Core-040419.pdf, 2019.

[37] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268, 2015.

[38] Stephen Gillen, Christopher Jung, Michael Kearns, and Aaron Roth. Online learning with an unknown fairness metric. In *Neural Information Processing Systems*, pages 2600–2609, 2018.

[39] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Neural Information Processing Systems*, pages 2672–2680, 2014.

[40] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[41] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330, 2017.

[42] Gurobi Optimization, LLC. Gurobi optimizer reference manual. `https://www.gurobi.com/documentation/8.1/refman.pdf`, 2019.

[43] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In *Neural Information Processing Systems*, pages 3315–3323, 2016.

[44] Úrsula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1944–1953, 2018.

[45] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[46] Sara Hooker. Moving beyond "algorithmic bias is a data problem". *Patterns*, 2(4):100241, 2021.

[47] Donghui Hu, Zhongjin Ma, Xiaotian Zhang, Peipei Li, Dengpan Ye, and Baohong Ling. The concept drift problem in Android malware detection and its solution. *Security and Communication Networks*, 2017.

[48] Christina Ilvento. Metric learning for individual fairness. *arXiv preprint arXiv:1906.00250*, 2019.

[49] David Ingold and Spencer Soper. Amazon doesn't consider the race of its customers. Should it? *Bloomberg*, 2016.

[50] International Warfarin Pharmacogenetics Consortium. Estimation of the warfarin dose with clinical and pharmacogenetic data. *New England Journal of Medicine*, 360(8):753–764, 2009.

[51] Roberto Jordaney, Kumar Sharad, Santanu K Dash, Zhi Wang, Davide Papini, Ilia Nouretdinov, and Lorenzo Cavallaro. Transcend: Detecting concept drift in malware classification models. In *USENIX Security Symposium*, pages 625–642, 2017.

[52] Marc Juarez, Samuel Yeom, and Matt Fredrikson. Auditing for distribution shift with black-box access. In submission, 2021.

[53] Christopher Jung, Michael Kearns, Seth Neel, Aaron Roth, Logan Stapleton, and Zhiwei Steven Wu. Eliciting and enforcing subjective individual fairness. *arXiv preprint arXiv:1905.10660*, 2019.

[54] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. Extremal mechanisms for local differential privacy. *Journal of Machine Learning Research*, pages 492–542, 2016.

[55] Kimmo Kärkkäinen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age. *arXiv preprint arXiv:1908.04913*, 2019.

[56] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2564–2572, 2018.

[57] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. An empirical study of rich subgroup fairness for machine learning. In *ACM Conference on Fairness, Accountability, and Transparency*, pages 100–109, 2019.

[58] Niki Kilbertus, Mateo Rojas Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding discrimination through causal reasoning. In *Neural Information Processing Systems*, pages 656–666, 2017.

[59] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. In *Innovations in Theoretical Computer Science*, pages 43:1–43:23, 2017.

[60] H. W. Kuhn and Bryn Yaw. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[61] Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. In *Neural Information Processing Systems*, pages 4066–4076, 2017.

[62] Jacob Leygonie, Jennifer She, Amjad Almahairi, Sai Rajeswar, and Aaron Courville. Adversarial computation of optimal transport maps. *arXiv preprint arXiv:1906.09691*, 2019.

[63] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

[64] Martina Lindorfer, Matthias Neugschwandtner, and Christian Platzer. MARVIN: Efficient and comprehensive mobile app classification through static and dynamic analysis. In *IEEE Computer Software and Applications Conference*, volume 2, pages 422–433. IEEE, 2015.

[65] Zachary Lipton, Julian McAuley, and Alexandra Chouldechova. Does mitigating ML's impact disparity require treatment disparity? In *Neural Information Processing Systems*, pages 8125–8135, 2018.

[66] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*, pages 3122–3130, 2018.

[67] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, 2015.

[68] Pranay K Lohia, Karthikeyan Natesan Ramamurthy, Manish Bhide, Diptikalyan Saha, Kush R Varshney, and Ruchir Puri. Bias mitigation post-processing for individual and group fairness. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2847–2851, 2019.

[69] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. ShuffleNet v2: Practical guidelines for efficient CNN architecture design. In *European Conference on Computer Vision*, pages 116–131, 2018.

[70] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. In *International Conference on Machine Learning*, pages 3381–3390, 2018.

[71] Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. *arXiv preprint arXiv:2104.10706*, 2021.

[72] Bernard Marr. How AI and machine learning are used to transform the insurance industry. *Forbes*, 2017.

[73] Jerzy Neyman and Egon Sharpe Pearson. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London*, 231 (694–706):289–337, 1933.

[74] Herve Nkiama, Syed Zainudeen Mohd Said, and Muhammad Saidu. A subset feature elimination mechanism for intrusion detection system. *International Journal of Advanced Computer Science and Applications*, 7(4):148–157, 2016.

[75] Judea Pearl. *Causality*. Cambridge University Press, 2009.

[76] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, pages 2825–2830, 2011.

[77] Michaël Perrot, Nicolas Courty, Rémi Flamary, and Amaury Habrard. Mapping estimation for discrete optimal transport. In *Neural Information Processing Systems*, pages 4197–4205, 2016.

[78] Kent Quanrud. Approximating optimal transport with linear programs. *arXiv preprint arXiv:1810.05957*, 2018.

[79] Joaquin Quiñonero-Candela, Masashi Sugiyama, Neil D Lawrence, and Anton Schwaighofer. *Dataset shift in machine learning*. MIT Press, 2009.

[80] Stephan Rabanser, Stephan Günnemann, and Zachary Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. In *Neural Information Processing Systems*, pages 1396–1408, 2019.

[81] Kaivalya Rawal, Ece Kamar, and Himabindu Lakkaraju. Can i still trust you?: Understanding the impact of distribution shifts on algorithmic recourses. *arXiv preprint arXiv:2012.11788*, 2020.

[82] Michael Redmond and Alok Baveja. A data-driven software tool for enabling cooperative information sharing among police departments. *European Journal of Operational Research*, 141(3):660–678, 2002.

[83] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *Information Processing in Medical Imaging*, pages 146–157. Springer, 2017.

[84] Vivien Seguy, Bharath Bhushan Damodaran, Rémi Flamary, Nicolas Courty, Antoine Rolet, and Mathieu Blondel. Large-scale optimal transport and mapping estimation. *arXiv preprint arXiv:1711.02283*, 2017.

[85] Alireza Shafaei, Mark Schmidt, and James J Little. A less biased evaluation of out-of-distribution sample detectors. *arXiv preprint arXiv:1809.04729*, 2018.

[86] Lloyd S Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2 (28):307–317, 1953.

[87] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy*, pages 3–18, 2017.

[88] Harvineet Singh, Rina Singh, Vishwali Mhasawade, and Rumi Chunara. Fairness violations and mitigation under covariate shift. In *ACM Conference on Fairness, Accountability, and Transparency*, pages 3–13, 2021.

[89] Supreme Court of the United States. *Griggs v. Duke Power Co.* 401 U.S. 424, 1971.

[90] Supreme Court of the United States. *Connecticut v. Teal*. 457 U.S. 440, 1982.

[91] Supreme Court of the United States. *Ricci v. DeStefano*. 557 U.S. 557, 2009.

[92] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[93] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114, 2019.

[94] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. MnasNet: Platform-aware neural architecture search for mobile. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

[95] Zilong Tan, Samuel Yeom, Matt Fredrikson, and Ameet Talwalkar. Learning fair representations for kernel models. In *Artificial Intelligence and Statistics*, pages 155–166, 2020.

[96] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. A detailed analysis of the KDD CUP 99 data set. In *IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–6. IEEE, 2009.

[97] Florian Tramèr, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, Jean-Pierre Hubaux, Mathias Humbert, Ari Juels, and Huang Lin. FairTest: discovering unwarranted associations in data-driven applications. In *IEEE European Symposium on Security and Privacy*, pages 401–416, 2017.

[98] Berk Ustun, Alexander Spangher, and Yang Liu. Actionable recourse in linear classification. In *ACM Conference on Fairness, Accountability, and Transparency*, pages 10–19, 2019.

[99] Rhema Vaithianathan, Emily Putnam-Hornstein, Nan Jiang, Parma Nand, and Tim Maloney. Developing predictive models to support child maltreatment hotline screening decisions: Allegheny County methodology and implementation. `https://www.alleghenycountyanalytics.us/wp-content/uploads/2018/02/DevelopingPredictiveRiskModels-package_011618.pdf`, 2017.

[100] Cédric Villani. *Optimal transport: Old and new*, volume 338. Springer Science & Business Media, 2008.

[101] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard Journal of Law & Technology*, 31(2):841–887, 2018.

[102] Xuezhi Wang and Jeff Schneider. Flexible transfer learning under support and model shift. In *Neural Information Processing Systems*, pages 1898–1906, 2014.

[103] Xuezhi Wang, Tzu-Kuo Huang, and Jeff Schneider. Active transfer learning under model shift. In *International Conference on Machine Learning*, pages 1305–1313, 2014.

[104] Karren D. Yang and Caroline Uhler. Scalable unbalanced optimal transport using generative adversarial networks. *arXiv preprint arXiv:1810.11447*, 2018.

[105] Samuel Yeom and Matt Fredrikson. Individual fairness revisited: Transferring techniques from adversarial robustness. In *International Joint Conference on Artificial Intelligence*, pages 437–443, 2020.

[106] Samuel Yeom, Anupam Datta, and Matt Fredrikson. Hunting for discriminatory proxies in linear regression models. In *Neural Information Processing Systems*, pages 4568–4578, 2018.

[107] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE Computer Security Foundations Symposium*, pages 268–282, 2018.

[108] Samuel Yeom, Irene Giacomelli, Alan Menaged, Matt Fredrikson, and Somesh Jha. Overfitting, robustness, and malicious algorithms: A study of potential causes of privacy risk in machine learning. *Journal of Computer Security*, 28(1):35–70, 2020.

[109] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *International Conference on World Wide Web*, pages 1171–1180, 2017.

[110] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pages 962–970, 2017.

[111] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *International Conference on Machine Learning*, pages 325–333, 2013.

[112] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

[113] Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. Domain adaptation under target and conditional shift. In *International Conference on Machine Learning*, pages 819–827, 2013.

[114] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[115] Zhe Zhang and Daniel B Neill. Identifying significant predictive bias in classifiers. *arXiv preprint arXiv:1611.08292*, 2016.

[116] Zhifei Zhang, Yang Song, and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2017.

[117] Han Zhao and Geoff Gordon. Inherent tradeoffs in learning fair representations. In *Neural Information Processing Systems*, pages 15675–15685, 2019.