Auditing the pymetrics Model Generation Process

Christo Wilson Alan Mislove Avijit Ghosh Shan Jiang
Khoury College of Computer Sciences, Northeastern University

Executive Summary

pymetrics is a startup that offers a candidate screening service (a.k.a. pre-employment assessment) to employers based on data and applied machine learning (ML). One of the core assertions pymetrics makes about their service is that they pro-actively de-bias ML models before deployment to comply with the U.S. Uniform Guidelines on Employee Selection Procedures (UGESP) [16]. pymetrics claims to use an outcome-based adverse impact testing process [48] where (1) candidate models are assessed for compliance with the UGESP four-fifths rule using minimum bias ratio as a metric and (2) models are retrained as necessary until a compliant model is identified [45, 43].

Building "fair" and "ethical" ML systems, as pymetrics claims to have done, necessitates more than software engineering — it requires a commitment to transparency and accountability so that employers and the public can be sure that the purported guarantees are faithfully implemented. This report presents one critical step towards transparency and accountability for pymetrics' candidate screening service: the results of an independent *algorithm audit* [54] of the source code that implements this system.

Scope. This algorithm audit was performed by academic researchers from Northeastern University in summer 2020, and focused specifically on pymetrics' claim that their candidate screening tool complies with their interpretation of the UGESP four-fifths rule. We (the academic auditing team) did not audit other pymetrics products or services; we did not investigate the ability of pymetrics' games to measure human capabilities or job performance; and we did not examine whether pymetrics' process or models obey alternative definitions of fairness in general or fairness for groups beyond those racial, demographic, and gender groups defined by the U.S. Equal Employment Opportunity Commission (EEOC). We elaborate on the scope of this audit in § 4.2.

Design. To perform the audit, pymetrics gave us access to the documentation and source code for their candidate screening service, as well as representative datasets. To maintain an arms-length relationship with pymetrics, we structured the audit as a grant to Northeastern University, with all payments made up-front and not contingent upon performance. Although pymetrics was aware of our audit, we did not inform them ahead-of-time about the methods we planned to use or the specific questions we planned to investigate. We retained the right to publicly publish the results of the audit, regardless of the outcome, with the exception of proprietary pymetrics source code, documentation, and datasets that were covered under a non-disclosure agreement (NDA). To demonstrate our commitment to full transparency as auditors, we have publicly released the contract, NDA, non-compete agreement, budget, and work plan for audit.[1]

Results. During our audit we investigated the following aspects of pymetrics' candidate screening tool and found the following results:

1. **Correctness.** We manually examined pymetrics' source code to determine whether it correctly implemented adverse impact testing as the UGESP four-fifths rule using the minimum bias ratio (a.k.a. impact ratio) metric as described in pymetrics' documentation. Additionally, we investigated whether fairness was assessed for the seven demographic categories defined by the EEOC (five racial and ethnic, two gender). We found that pymetrics source code correctly implemented these algorithms for the seven targeted groups.

[1]https://cbw.sh/audits.html

2. **Direct Discrimination.** We examined pymetrics source code and found that their model training process did not rely on demographic information from game players. Rather, demographic data was only used when performing post-training adverse impact testing of models.
3. **De-biasing Circumvention.** We attempted to circumvent pymetrics' adverse impact testing process using maliciously crafted datasets, i.e., to try and trick pymetrics' system into producing a biased model (and falsely believing that it was fair). Our attacks were unsuccessful: all control flow paths in the source code lead through the adverse impact tests. Our attacks either resulted in pymetrics' code being unable to train a fair model, or the code was able to mitigate our biased data and train an unbiased model. Either outcome meant that our attack had failed.
4. **Sociotechnical Safeguards.** Pymetrics' process for producing models involves human intervention, which raises the issue that human errors may subvert fairness guarantees. We observed that pymetrics has implemented safeguards against human error, in the form of a compliance checklist that two pymetrics data scientists must complete before any given model is released into production.
5. **Sound Assumptions.** Preprocessing datasets to make them suitable for ML may introduce subtle biases into the data. We extensively analyzed a key preprocessing step in pymetrics' ML pipeline: missing data imputation. While we found that pymetrics' current imputation algorithm does have differential impact on game players from different demographic groups, these issues ultimately had no substantive impact on pymetrics' adverse impact testing process or the fairness guarantees of trained models.

In summary, we are comfortable stating that pymetrics passed this audit, subject to the qualifications and limitations we state in § 4.2 and § 4.4. Based on extensive examination and testing of their source code, we can confirm that pymetrics' claims about the process they use to conduct adverse impact testing of models is implemented faithfully in their source code, and that it does result in models that conform to the four-fifths rule with respect to gender and ethnicity/race (as defined by the corresponding EEOC categories).

1 Introduction

Discrimination is a longstanding and pernicious problem in hiring. Despite years of legislation and regulation [57], as well as the development of techniques like implicit bias training and "blind" recruitment, studies continue to find that discrimination along demographic lines is systemic [47].

The application of machine learning (ML) techniques and big data to recruitment and hiring have brought a new dimension to concerns about bias, equality, and equity in the space. Employers are eager to capitalize on the potential for ML to bring scale and uniformity to the costly process of recruitment. Additionally, there is potential, or at least the perception of potential, for ML to remove human biases from the hiring process by minimizing the number of human beings involved in the process. However, computer scientists have demonstrated that biases can easily creep into ML systems [21, 41, 61, 36, 20, 32], raising concerns that automated hiring may end up replicating and entrenching existing inequities.

pymetrics is a startup that offers a candidate screening service (a.k.a. pre-employment assessment) to employers based on data and applied ML. One of the core assertions pymetrics makes about their service is that they pro-actively de-bias ML models before deployment to comply with the U.S. Uniform Guidelines on Employee Selection Procedures (UGESP) [16]. pymetrics claims to use an outcome-based model debiasing process [48] where (1) candidate models are assessed for compliance with the UGESP four-fifths rule using minimum bias ratio as a metric and (2) models are retrained as necessary until a compliant model is identified [45, 43].[2]

While pymetrics has vocally adopted "ethics" and "fairness" as differentiators for their service, achieving these ideals necessitates more than promises — transparency and accountability are also required, to ensure that these promises are being kept. pymetrics took the first step in this direction by open-sourcing some of the code they use to bias test models, opening it up to independent scrutiny [46].

This report presents the next step towards transparency and accountability for pymetrics' candidate screening service: the results of an independent *algorithm audit* [54] of the source code that implements this system. This algorithm audit was performed by academic researchers from Northeastern University in summer 2020, and focused specifically on pymetrics' claim that their candidate screening tool complies with the UGESP four-fifths rule. We, the Northeastern team, were selected for this audit because we have extensive experience auditing algorithms used by large platforms like Google [23, 33, 51, 53, 26, 52], Uber [10, 28], Amazon [11], and Facebook [59], as well as employment-related companies like Fiverr, Taskrabbit [25], Monster, CareerBuilder, and Indeed [9].

To perform the audit, pymetrics gave us access to the documentation and source code for their candidate screening service, as well as representative datasets. In § 4, we discuss the design of our audit, its scope, and the steps that we took to maintain an arms-length relationship with pymetrics. In § 5 we present the results of the audit, including the specific issues that we investigated, the methods we used, and our findings. We conclude in § 6 with a summary of our findings and opportunities for further investigation.

2 Background

We begin by presenting context for our audit, starting with concerns about the use of ML in hiring and followed with a brief introduction to the practice of algorithm auditing.

2.1 Fairness in Algorithmic Hiring

Since at least the mid-1990s, critical scholars have been raising alarms about the potential for computer systems to embed, entrench, and compound social biases [22]. These voices have grown louder and the concerns more acute as data-driven ML systems have seen increased adoption [6]. With respect to ML, social biases may creep into systems through a variety of vectors, such as biases in training data or cleaning methods, poor choice of learning objective, and mismatch between technical and sociotechnical context.

The adoption of ML techniques in the domain of hiring is particularly contentious. On one hand, discrimination in hiring driven by human biases is a long standing and widespread problem [47]. From this perspective, using ML to evaluate job seekers has the potential to remove humans and their biases from the

[2]Or, if no compliant model can be found, pymetrics abandons the training process and no model is deployed.

hiring process, potentially leading to more equitable outcomes. On the other hand, there is no reason to assume a priori that technical systems in the hiring domain will automatically be "objective," "neutral," or "bias-free." Indeed, algorithm audits of gig-economy marketplaces and traditional job boards that rank candidates based on their resumes have uncovered biases in these systems along gender and race lines [25, 9].

As startups have emerged that apply ML to the hiring process, including pymetrics, scholars have begun to investigate the legal, conceptual, and practical space in which they operate. Raghavan et al. surveyed publicly available information about 18 startups offering *pre-employment assessment* systems to document their practices and map them into the law and policy space, especially with respect to claims about compliance with the UGESP's four-fifths rule [48]. Ajunwa and Kim both present extensive taxonomies of the ways that bias may emerge in ML-based hiring systems and map these to U.S. legal doctrine [31, 1].

Title VII of the U.S. Civil Rights Act of 1964 distinguishes between two forms of discrimination that may impact hiring processes: *disparate treatment* and *disparate impact* [57]. The former refers to cases where people are directly discriminated against based on legally protected attributes, such as race and gender. In the ML context, avoiding disparate treatment is often operationalized as a prohibition against the use of protected attributes as input features for models [9, 1]. Disparate impact refers to cases where a facially neutral process still produces substantially different outcomes for people that are correlated with legally protected attributes. One famous example of disparate impact is "red lining:" by using U.S. Zip Code as a factor in determining whether to lend money, banks where able to discriminate against minority applicants without explicitly taking race/ethnicity into account [42]. With respect to ML, there are a variety of de-biasing techniques that have been developed to ensure that models do not produce disparate impact [21, 41, 61, 36], although scholars have found that not all of these fairness objectives are mathematically compatible in practice [20, 32]. We examine both disparate treatment and disparate impact in our audit of pymetrics.

2.2 Algorithm Auditing

Raji et al. write that "audits are tools for interrogating complex processes" [49]. With respect to modern sociotechnical systems, Sandvig et al. motivate the need for algorithm audits as a means "to investigate normatively significant instances of discrimination involving computer algorithms operated by Internet platforms" [54]. While some have likened algorithm auditing to reverse engineering by outsiders, in that the goal is to make "black-box" systems more transparent regardless of the system creator's intent [13], this conceptualization has since been expanded to include audits carried out by ethically and morally-conscious insiders [49].

There is a growing body of algorithm audits carried out by academics and investigative journalists assessing a variety of systems for a diverse set of harms. This includes examining broad classes of systems like search [39, 14, 34, 30, 23, 33, 14, 53, 51, 26, 52], e-commerce [24, 11], news recommendation [29, 5], online advertising [56, 58], maps [55], ridesharing [10, 28], online reviews and ratings [17, 18], natural language processing [8], and recommendation [27]. Some audits have specifically focused on algorithms in high-stakes contexts like facial recognition [7], predictive policing [2, 15, 37], access to housing [3], and child protective services [12].

In the absence of regulation or accepted best-practices, recent scholarly work has attempted to define a process for algorithm auditing. Raji et al. developed a six step process for auditing that we roughly follow in our audit of pymetrics [49]. This process includes scoping (see § 4.1), mapping (which involves interviewing stakeholders, see § 4.3.3), artifact collection (again, see § 4.3.3), testing (§ 5), and reflection (of which a large part is generating reports like this one).

Raji et al. draw a distinction between *internal* and *external audits* [49]. In Raji et al.'s parlance, an internal audit of (e.g.,) pymetrics would be conducted by pymetrics employees in parallel with the development of their ML systems, while an external audit would be conducted by experts with no association to pymetrics and no privileged access to pymetrics' systems (e.g., by using a public-facing API). The audit we present in this study does not fall into these paradigms: we are not employees of pymetrics and we were not present during the development of their systems, yet we were given privileged access to pymetrics source code and documentation (see § 4.3.3). Thus, we refer to this endeavor as a *cooperative audit* as it involves cooperation between internal and external actors.

Sandvig et al. introduced five designs for conducting algorithm audits [54]. Our study of pymetrics corresponds most closely with a *code audit* in the Sandvig et al. taxonomy since we directly examined pymetrics' source code and datasets. However, Sandvig et al. assume that source code will be publicly available so that a key precept of classic audit study design can be maintained: that the audited party not be aware of the audit. This assumption is not true in our case, since our audit of pymetrics was cooperative. As we discuss in § 4.3.4, we undertook other steps to insure our independence from pymetrics.

Several tools have been developed by academics to facilitate auditing of black-box ML models, including LIME [50] and SHAP [38]. These advanced statistical tools were not necessary for this audit, since (1) training data and source code were available to us, and (2) pymetrics uses interpretable models (see § 3.2).

3 About pymetrics

In this audit we focus on the candidate screening (a.k.a. pre-employment assessment [48]) product offered by pymetrics. Pymetrics is a startup founded in 2013 that offers a number of services in the context of employment. Unlike job board services like Monster.com or Indeed, pymetrics is not a marketplace where employers post jobs or job seekers post resumes. Rather, pymetrics uses applied ML to make predictions about job seekers.

Pymetrics' candidate screening service is designed to filter out applicants applying for jobs before the interview stage while simultaneously attempting to avoid disparate impact by abiding by the UGESP's four-fifths rule with respect to protected demographic groups. At a high-level, pymetrics' candidate screening service can be summarized as follows [45]:

1. An employer contracts with pymetrics to develop and deploy a predictive model for candidate screening. We refer to these employers as *clients*.
2. A *job analyst* from pymetrics surveys the client to understand the *target role* (e.g., the job description, seniority-level, etc.) and the metrics that the client uses to assess job performance in that role [44]. This information informs the selection of unlabeled training data for the predictive model (among other things).
3. The client has incumbent employees in the targeted role play pymetrics' suite of games (which we describe further in § 3.1). The client also gives existing job performance data about these incumbents to pymetrics. Together, this performance and gameplay data serves as the labeled training data for a predictive model.
4. A pymetrics data scientist uses an internal pymetrics tool to develop a predictive model for the client. Unlabeled training data and a held-out set of data for adverse impact testing are selected by the data scientist, and used alongside the labeled training data from the client to train ML models. These models are evaluated for predictive performance and compliance with the UGESP's four-fifths rule.[3]
5. Pymetrics deploys the best-performing predictive model that meets the fairness criteria. Job seekers who apply for the targeted role at the client are asked to play the pymetrics' suite of games. Based on this gameplay data, the model predicts which candidates have similar attributes to the clients' high-performing incumbent employees. Information about high-scoring job seekers are sent to the client, who may then apply additional filters (e.g., resume screening) and begin interviewing candidates.
6. Pymetrics performs longitudinal analysis of the predictive model. This includes re-evaluating whether fairness criteria are being met with respect to the pool of job seekers that have applied for this role, and studying the job performance of candidates who were hired.

3.1 Data Sources

Pymetrics' candidate screening service relies on a variety of data sources to train and evaluate ML models. The primary data source is a core set of twelve games that are derived from peer-reviewed psychological studies.[4] These games are purported to assess intrinsic mental qualities of individuals — the games are not

[3]If the pymetrics data scientist is unable to train a suitable model (i.e., due to insufficient performance or failure to meet fairness criteria) then the job analyst is reengaged to help the client refine the definition of the targeted role, select better incumbent employees, and/or develop better assessment metrics for incumbent performance).

[4]Pymetrics has recently introduced an additional set of four numerical and logical reasoning games that clients may ask pymetrics to incorporate into their models. As these games are new, as of 2020, and not yet widely played, we ignore them in this study.

meant to be won or lost, but rather to surface information about players based on how they play. Each game produces a number of results per player, which pymetrics refers to as *traits* (a.k.a. features in ML parlance). These games are available on the web, iOS, and Android, are translated into several languages, and have built-in accommodations for players with color-blindness and/or dyslexia.

After players complete the pymetrics games, they are asked to take an optional demographic survey. The survey asks for their gender (male/female) and ethnicity/race (Asian/Black/Hispanic/White/two-or-more groups). These categories correspond to those delineated by the EEOC for adverse impact testing. Pymetrics reported to us that over 75% of players complete the demographic survey [40]. This data is used to construct held-out sets of data that are used for adverse impact testing.

Finally, as we describe in § 3, pymetrics asks clients to provide performance data about incumbent employees. The metrics used to assess performance and the interpretation of this data vary across clients and roles. Ultimately, pymetrics job analysts and data scientists work together to distill this performance data to a single feature vector that can be used as the labeled data to train predictive models.

3.2 Model Training

This section has been redacted by the authors at the request of pymetrics. The original text contains a detailed explanation of the pymetrics model training pipeline that reveals proprietary information about the system. Per Northeastern's contract with pymetrics, proprietary information of this nature is covered by non-disclosure rules. The contract is available here.

We direct readers who are interested in a sanitized, non-proprietary version of this information to our peer-reviewed, co-authored manuscript that appeared in the ACM Conference on Fairness, Accountability, and Transparency (FAccT).

3.3 Adverse Impact Testing

At each step of model training, the trained model is evaluated for compliance with the four-fifths rule using the minimum bias ratio metric. According to pymetrics' documentation [45]:

> All pymetrics models are proactively de-biased before they are deployed for selection purposes. Models must pass the 4/5ths rule when performance is compared across demographic groups during model build (see UGESP, EEOC et. al., 1978). Bias is assessed by using a withheld set of users for whom demographic information is known, called the bias set. The bias set contains users from each EEOC-defined race, ethnicity and gender group. De-biasing on additional demographic groups may be requested and implemented depending on the geographic relevance and legal requirements of the use case.
>
> The bias set is tested against each pymetrics model. The proportional pass rates of the highest-passing demographic group are compared to the lowest-passing group for each demographic category (gender race, and ethnicity). This proportion is known as the min bias ratio (also referred to as the Impact Ratio).
>
> . . .
>
> Bias testing is a threshold measurement: a model either passes or does not pass pymetrics checks for bias. Even beyond this threshold however, where concurrent criterion-related validity metrics are equal, the data scientist responsible for building the model will select a model with least estimated bias.

Pymetrics' guide for fairness testing procedures provides more specific details of how adverse impact assessment is performed [43]:

> Pymetrics conducts fairness analyses (i.e., adverse impact analyses) by comparing pass rates across demographic groups (i.e., gender, race) using the pymetrics recommendation level aligned with the agreed upon implementation (e.g., passing Highly Recommend and Recommend to the next round in the recruiting process) and using tests of both statistical significance (Chi Square and Fisher's Exact test of independence; $p < .05$ threshold) and practical significance (4/5ths rule impact ratio). Both approaches are outlined by the Uniform Guidelines for Selection Procedures (UGESP) and together serve as the gold standard for adverse impact testing.

Pymetrics uses trained models to score job seekers — this is done to place each job seeker into "Highly Recommended," "Recommended," and "Not Recommended" categories. Pymetrics sets score thresholds to delineate the boundaries between categories. This categorization raises a complication with respect to measuring compliance with the four-fifths rule: the pass rate for a given demographic group may vary based on the chosen threshold. According to pymetrics' documentation they address this issue in the following manner [45]:

We do this through a de-biasing process whereby models are checked for bias at both the 50% and 70% fit percentile thresholds (delineating "do not recommend," "recommend," and "highly recommend" score bands)

Pymetrics claims that only models that obey the above fairness criteria are deployed. Determining whether this is true is the focus of our audit.

4 Designing the Audit

In this section we discuss the design of our audit, including what we examined, what we did not examine, the baseline requirements for conducting the audit, and how we managed our relationship with pymetrics.

4.1 Scope

The primary focus of our audit is pymetrics' claim that their model training process produces models that abide by pymetrics' interpretation of the UGESP's four-fifths rule [16]. Towards this end, we examined documentation and source code that implements pymetrics' candidate screening product, which relies on data from twelve online games to produce labeled and unlabeled data for training predictive models. We conducted our audit in summer 2020.

Pymetrics' documentation elaborates on the specific steps they claim to use to perform adverse impact testing on trained models [43]:

Adverse impact testing will follow these steps:

1. Subset data to include only those that make up the position/location of interest
2. Subset data to exclude any demographic group making up less than 2% of the available data for the position/location of interest
3. Calculate selection ratios for each demographic
4. Find largest selection ratio to serve as comparison
5. Calculate impact ratios by dividing each demographic's selection ratio by the comparison group's selection ratio
6. Significance test the differences in pass/fail rates using chi-square and fisher's exact test
7. Repeat steps 5 & 6 for each comparison
8. Repeat steps 2 - 7 for each demographic variable (e.g., gender, race)
9. Repeat steps 1 - 8 for each position/location combination

At a high-level, our goal is to assess whether these steps accurately describe the pymetrics' implementation, and whether these steps are sufficient to ensure that the resulting models are fair to job seekers as defined by the four-fifths rule. During the audit, we focused on the following specific questions:

1. **Correctness.** Pymetrics' documentation describes their process for performing adverse impact testing on trained models before they are deployed. *Does the model training source code correctly implement adverse impact testing as the four-fifths rule using the minimum bias ratio (a.k.a. impact ratio) metric as described in the documentation? Is fairness assessed for the seven demographic categories defined by the EEOC (five racial and ethnic, two gender)?*
2. **Direct Discrimination.** Using demographic data as training features for models can be construed as a form of direct discrimination. This motivates us to ask *do trained models use demographic data directly as input, or is demographic data only used for post-training adverse impact testing?*
3. **De-biasing Circumvention.** There are numerous examples of deployed ML-based systems that had their safety systems subverted by clever and malicious users [35, 60]. We have made similar observations in our previous audits of online platforms [11, 25, 59]. These experiences motivate us to ask *is there any way for training data that is erroneously corrupted or intentionally biased to somehow avoid the adverse impact tests, thus resulting in an unfair model being released?*
4. **Sociotechnical Safeguards.** Pymetrics' process for producing models involves human intervention, which raises the issue that human as well as technical errors may subvert fairness guarantees. *Does pymetrics have checks in place to ensure that human errors (either benign or malicious) do not result in an unfair model being released?*

5. **Sound Assumptions.** Using ML is never as simple as loading data and inputting it into a training algorithm. Data must be qualitatively assessed as being fit-for-purpose, and then be preprocessed and transformed to prepare it for analysis. This process concretizes assumptions about the data that may ultimately impact the resulting trained models and the adverse impact assessment. *Are there assumptions about data and data preprocessing baked into pymetrics' model training process that could cause the adverse impact assessment to fail or mislead?*

4.2 Out of Scope

Just as important as understanding what we **are** auditing is understanding what we **are not** auditing. This point is critical for properly contextualizing our audit and bounding what was inside and outside our scope. In particular, our audit did not cover the following aspects of pymetrics' products and business.

- Prior to conducting the audit, we agreed with pymetrics that we would not question their choice of fairness objective (the UGESP four-fifths rule) or fairness metric (minimum bias ratio). Although there are many other potential fairness objectives and metrics [21], including others designed to prevent disparate impact [41, 61, 36], pymetrics' chose their existing objective and metric based on what they felt was most appropriate in the context of their business, i.e., candidate screening. This objective and metric were proposed by the relevant U.S. regulators themselves.
- Similarly, we agreed not to question pymetrics' choice of race, ethnicity, and gender categories that they evaluate for fairness since these categories are delineated as protected by the EEOC. Further, we agreed to not evaluate fairness for intersectional groups (i.e., combinations of demographic categories like Black males or Asian females) since they are not considered protected by the EEOC.
- We only audited pymetrics' game-based candidate screening product. We did not audit other pymetrics products and services.
- We did not investigate the ability of pymetrics' games to measure human capabilities, whether those capabilities map to job performance, or whether other assessment methods would be superior in some respect (fairness or accuracy). As computer scientists, evaluating these aspects of the pymetrics system are beyond our capabilities. Additionally, we do not comment on the rationality and ethics of using these measures to evaluate a candidate's suitability for hire.
- Pymetrics recently started offering an additional suite of numerical logic and reasoning games. We did not have access to datasets that included data from these games, so we cannot comment on their specific impact to fairness. That said, as we discuss in § 5.1, the control flow in the pymetrics source code ensures that all models eventually must pass fairness checks, regardless of whether the model includes or does not include data from these additional games.
- Pymetrics performs post-training adverse impact testing on models using a held-out set of data [43, 48]. Prior to conducting the audit, we agreed with pymetrics that we would not question their choice to use post-training testing. While pre-training [19] and during-training model de-biasing methods [61] exist, they require that training data include complete demographic information, which is not always available in employment contexts.[5]
- During our audit, we did not focus on evaluating or maximizing the predictive performance of pymetrics' models. Rather, fairness was our main concern. That said, during our testing, we did obey the minimum baseline predictive performance requirements that pymetrics demands of all their models. See § 5.3.3 for further details.
- We did not audits pymetrics' process for performing annual adverse impact back testing on deployed models [45].
- We did not examine pymetrics' cybersecurity posture, e.g., we did not perform penetration tests or red team reviews. We did not attempt to become a pymetrics client, play their games while posing as an employer or a job seeker, have any contact with pymetrics employees outside the narrow confines of this audit, or attempt to conduct insider-attacks given our privileged access to pymetrics systems, data, and employees.

[5]For example, an employer may not be willing to divulge demographic information about incumbent employees.

- We did not examine pymetrics posture with respect to data privacy or compliance with laws like Europe's General Data Protection Regulation (GDPR), the California Consumer Privacy Act (CCPA), the U.S. Children's Online Privacy Protection Act (COPPA), etc.

4.3 Requirements

To fully and completely answer the questions posed above in a way that the public can trust necessitated that we establish a number of requirements for our audit. Northeastern University and pymetrics signed a contract in March 2020 agreeing to the scope of the audit and these requirements before we commenced the audit. Here, we outline the requirements of our audit.

4.3.1 Transparency

One of the foremost issues we identified heading into the audit was how to preserve our objectivity and trustworthiness. These properties are essential, both from the audit design perspective (i.e., are we asking the right and tough questions) and from the public reporting perspective (i.e., will people believe the results of the audit).

To promote these properties, we adopt a stance of transparency. The contract between Northeastern University and pymetrics, the audit and data sharing protocol agreed to by the the audit team and pymetrics, the non-compete signed by the lead auditor (Christo Wilson), and the project budget are publicly available online for anyone to inspect.[6] As stipulated in the contract, the only facets of the project covered by non-disclosure rules include source code, data, and internal manuals from pymetrics. Other than these pieces of proprietary information, the audit team retained the right to speak and publish about the audit, up to and including specific results that might reflect negatively on pymetrics.

As specified in the contract, the audit team agreed to a policy of *responsible disclosure* with respect to problems or issues uncovered during the audit. Specifically, the audit team agreed not to publicly discuss problems or issues uncovered during the audit until pymetrics was given 30 days to privately remediate the issues. This policy was modeled on industry standard disclosure practices from the realm of cybersecurity.

4.3.2 Remuneration

On the subject of objectivity and trustworthiness, another fraught issue that we identified was remuneration, i.e., should pymetrics pay for the audit, and if so, how? On one hand, we as auditors want to avoid real and perceived conflicts of interest. Accepting payment for the audit immediately raises legitimate concerns about our objectivity.

On the other hand, performing a pro-bono audit raises the issue of setting an unrealistic, exclusionary precedent. To our knowledge, there have been very few cooperative algorithm audits between companies and independent investigators that have been made public. It is our sincere hope that our audit of pymetrics will serve as model for more cooperative audits in the future. With this framing in mind, if we established a precedent that only pro-bono audits are sufficiently objective, this could severely limit who is able to serve as an auditor in the future. In the absence of funding from neutral sources (e.g., grants from government, foundations, or endowments), many academics, investigative journalists, etc. may not have the financial means to support themselves while conducting audits. In a world increasingly permeated by algorithmic systems, we argue that we need as many skilled auditors as we can get.

Further, completing the audit pro-bono raises issues of exploitation. Auditing is challenging work that requires a high-level of expertise. Especially when graduate student labor is involved, as it was during this audit, setting a precedent of work without compensation exacerbates pre-existing power imbalances in academia.

We reached out to a number of people in the academic community who are well versed in the study of online platforms for advice on how to approach the issue of remuneration. Based on their extremely helpful guidance, we decided to accept payment for the audit subject to a number of constraints. *First*, the payment was structured as a grant to Northeastern University, not as direct payment to the auditors as independent contractors. This added a layer of institutional oversight to the project. *Second*, payment was

[6] https://cbw.sh/audits.html

delivered in installments over the project period in summer 2020 and was not conditioned on performance. All payment was received before we delivered the findings of the audit to pymetrics, thus mitigating concerns that payment could be conditioned on positive audit results. As noted above, the contract laying out these stipulations and the project budget are both publicly available.

4.3.3 Access and Materials

To complete this audit, we were given extraordinary access to pymetrics. At the outset, we spent a day with pymetrics employees learning about the company, their data science pipeline, and how they approach fairness issues, as well as demos of how pymetrics data scientists use their internal tools to train and evaluate models. During this "onboarding" we were also given copies of internal pymetrics documents that present, among other things, a technical overview of their candidate screening product [45] and specific details about their fairness testing procedure [43]. Pymetrics reported to us that these documents are made available to (prospective) pymetrics clients under a non-disclosure agreement.

pymetrics gave us access to source code for their candidate screening product. At a high-level, the source code encompasses a "template" Jupyter notebook that is used by pymetrics data scientists, along with associated Python modules. The notebook implements the data scientist-facing process of producing a predictive model for a specific client, including loading data, cleaning data, checking the raw data for conformance to baseline suitability criteria, initiating the Beam search for an accurate and fair model, performing adverse impact testing, performing any manual tweaks to the model, and finally packaging the model for deployment. According to pymetrics, the Python modules implement specific algorithms that are generally constant across all client engagements, such as biased SVM, Beam search, and the minimum bias ratio metric.

We were given eight notebooks in total:

- **Blank Template.** One notebook was "blank," in the sense that it had not been filled out by a data scientist. In other words, this is how the template notebook appears to a pymetrics data scientists that is beginning a new client engagement from scratch — it contains scaffolding code and processes but no specifics.
- **Representative Samples.** Six notebooks were sampled from recent and completed client engagements. These notebooks had each been filled out by a pymetrics data scientist and produced a model that ultimately went live into production. Five of these notebooks were selected uniformly at random by pymetrics from client engagements that had occurred within the six months preceding the audit. The sixth notebook was chosen because it came from a recent engagement where the client requested extensive changes to the adverse impact testing process.
- **Complete Engagement.** The final notebook also came from a completed client engagement, and included the associated datasets that were used to train and evaluate the models. The bulk of our attention during this audit was on this "complete" notebook and its associated data.

The seven completed notebooks were anonymized to remove specific references to the client companies. The data from the complete engagement was pseudonymous: it contained no personally identifiable information (PII), but gameplay and demographic data was associated with individual game players.

All of these notebooks and data were uploaded to a virtual machine that was provisioned by pymetrics and hosted on Amazon Web Services. The audit team performed all analysis within this virtual machine. We agreed to confine our activities to within this virtual environment to obviate pymetrics' concerns about their proprietary code and data being leaked.

Finally, after the completion of our analysis, we observed a live demonstration of a pymetrics data scientist training, testing, and deploying a model into production use. This demonstration allowed us to confirm that the notebooks we analyzed, and the process they concretize, are the same as what pymetrics uses in production.

4.3.4 Independent Testing

An essential principle of auditing is that, to the greatest extent possible, the subject should not know the manner in which they are being evaluated, or be allowed to dictate the tests that will be conducted. In keeping with this principle, we did not inform pymetrics of the tests or testing methods we planned to use before commencing the audit. The extent of pymetrics' knowledge prior to conducting the audit was (1) that the audit was taking place, (2) that we would be evaluating the model training and testing portion of their game-based candidate screening product, and (3) that we might employ "fake" or synthetic data in our testing.

We maintained this posture of secrecy throughout the course of the audit. During our initial onboarding with pymetrics staff we made sure to ask only general questions that would not reveal the focus of our testing or our methods. Similarly, at several points during the audit we required technical assistance to run pymetrics code, as well as additional datasets that were not provided initially. During these interactions were also opaque, and did not elaborate on why we wanted things or our specific motivations. The pymetrics team honored our requests and did not attempt to extract information about the status of our testing.

4.3.5 Deliverables

As outlined in our contract with pymetrics, the required deliverables from this audit were this report, as well as any source code and data we produced during the course of the audit.

4.4 Limitations

As with any scientific study, it is critical to be forthright about the limitations of our audit study.

First, we operated under an assumption of good faith on the part of pymetrics. We assume that documentation, source code, and data that we received are representative of the actual, deployed systems and data used by pymetrics. Given that pymetrics agreed to full transparency of this audit, we have no reason to doubt their sincerity. Additionally, we observed a live demonstration of a model being trained, tested for adverse impact, and deployed by a pymetrics data scientist. This demonstration allowed us to confirm that the "production" source code and process matched the one we audited.

Second, our audit examined pymetrics' fairness claims and source code in summer 2020. Pymetrics' claims, source code, and products may change over time, and we cannot make statements about these new and modified systems, nor can we make any statements about pymetrics' practices and guarantees before we ran our audit.

Third, part of pymetrics' business model is that they may customize their model training and adverse impact assessment process for specific clients. This may include adding or removing specific game traits, altering the demographic groups that are assessed for fairness, and even swapping out the four-fifths fairness metric for an entirely different fairness criteria. Since we cannot audit these bespoke processes or the models they produce we make no statements about them. Our audit results are only representative with respect to pymetrics' "baseline," non-customized model training and adverse impact assessment process.

5 Results

In this section we present the results of our audit. We organize our discussion around the six questions given in § 4.1 using the source code and data introduced in § 4.3.3.

5.1 Overall Implementation

We begin by addressing three questions:

1. **Correctness.** Does pymetrics' source code faithfully implement the four-fifths rule via the minimum bias ratio metric, using the process described in their documentation?
2. **Direct Discrimination.** Do models trained using the pymetrics source code directly incorporate demographic features, and/or do the models take demographic features as direct input?
3. **De-biasing Circumvention.** Is there a way to manipulate the input data to the pymetrics source code in such a way that the fairness checks are circumvented?

To answer these questions, we manually examined the source code provided by pymetrics. This includes the Jupyter notebook that is used by pymetrics data scientists, as well as the custom Python modules that were implemented by pymetrics. As we described in § 4.3.3, pymetrics gave us six completed notebooks in addition to the "blank" template notebook — our analysis is based on these notebooks. With the exception of one notebook that was heavily modified to suit a particular client, the remaining five notebooks had consistent source code and use of custom modules.

With respect to correctness, we found that pymetrics' source code does implement the four-fifths rule using the minimum bias ratio metric, with the seven considered groups being males, females, Whites, Blacks, Hispanics, Asians, and people who identify with ≥ 2 racial or ethnic groups. The Jupyter notebook "template" that pymetrics uses for typical client engagements guides the data scientist through the process of: loading data (the *in group* from the client, a randomly sampled *out group* from people who have played the games in the past, and a randomly sampled *bias group* of people who played the games in the past **and** completed a demographic survey); cleaning the data and preparing it for analysis;[7] executing tree searches for potentially biased traits and unbiased models; manually tweaking the top performing model, if necessary; and packaging the model for deployment. Along the way, the randomly sampled *out* and *bias group*, intermediate data, and model performance data are logged to facilitate debugging and reproducibility. Before model training, the data scientist is presented with correlations between game traits and demographics, and is given the option of manually excluding traits that are obviously biased. After model training, the performance of the "best" models are presented to the data scientist, where "best" is defined as meeting minimum criteria for AUC, accuracy, recall, and precision, **and** passing the four-fifths rule.

Digging deeper into the Python module source code, we examined the code paths that perform the Beam search for unbiased models. This is where the code for calculating minimum bias ratio and using it to guide feature exploration resides. We found no issues with this code.

With respect to direct discrimination, we confirmed that players' demographic information is not used as features for model training. The *in group* and *out group* datasets are used for model training, and they do not contain demographic information. Only players in the *bias group* have corresponding demographic information, and the *bias group* is only used for trait and model evaluation, e.g., calculating the pass rates for players of varying demographics with respect to a given model.

With respect to circumventing fairness checks, we were unable to find a way to produce a biased model that was not flagged as such by the analytical tools in the Jupyter notebook. For the purpose of this testing we assume the following *threat model*: a pymetrics client may arbitrarily manipulate the *in group* dataset that they supply to pymetrics. Specifically, the client can change the size of the *in group* as well as produce arbitrary gameplay data.[8] By manipulating these aspects of the *in group* data, the malicious client's goal is to get pymetrics to unwittingly deploy a biased model.

Our threat model is intentionally abstract to cover a wide range of potential malicious behaviors. In practice, a pymetrics client could, hypothetically, engage in several types of specific malicious behavior. One possibility is that a client could supply an *in group* dataset that contains information from a demographically homogeneous group of employees, e.g., a bunch of white men. Another possibility is that a client could lie by (1) having a single employee play the pymetrics' games 50 times and then (2) supplying pymetrics with fabricated performance data that misrepresents the single employee as 50 unique individuals. In either attack scenario, the hypothetical malicious client's goal is to get pymetrics to unwittingly produce and deploy a biased model that reifies the client's existing, non-diverse workforce.

In our testing, we were unable to circumvent the fairness checks in pymetrics source code by manipulating *in group* data. All control flow paths in the Jupyter notebook eventually arrive at the tree search that trains models and evaluates them against the four-fifths rule. It is possible to get the model search to fail, e.g., by supplying an *in group* that is too small or has wildly unbalanced gameplay data, but these failures would (presumably) be immediately recognized by the data scientist — the model production and deployment process cannot continue until a compliant model is produced. More subtle attacks also failed, such as supplying *in group* data that we synthetically crafted to over-sample players with specific demographic

[7]We discuss data cleaning and preparation in more detail in § 5.3.

[8]Note that this threat model is unrealistically strong. In practice, it would be very difficult for a client to produce arbitrary gameplay data, since they would either need to train human beings to play the pymetrics games in very specific ways, or write software to emulate a human and play the games. Further, pymetrics' clients work closely with a human job analyst from pymetrics to select incumbent employees and fairly evaluate their performance. A malicious client would need to lie to the job analyst in addition to producing manipulated data.

traits (e.g., White males). Depending on the construction of the synthetic data, model training was either unable to generate a model that met the four-fifths rule and failed immediately, or model training was able to generate a model that obeyed the four-fifths rule. Both outcomes are failures from the perspective of the attacker.

5.2 Human Oversight

The next question we examine concerns **sociotechnical safeguards**. It should be apparent at this point that pymetrics' process for generating models requires a substantial amount of human involvement. Rather than being a fully automated process where input data goes in and a model comes out and goes straight into deployment, models at pymetrics are crafted by hand. Involving data scientists has benefits: they can notice and correct issues during model building, and tweak models to better support the unique needs of clients.

However, human involvement also raises concerns. Could a malicious or negligent data scientist produce and deploy a biased model? There are no programmatic constraints in the Jupyter notebook that prevent a data scientist from training a model, failing to check it for bias, and uploading it to pymetrics' back-end system that hosts trained models. Further, building such programmatic constraints would be extremely difficult and perhaps impossible — this is a fundamental tradeoff that comes with empowering human data scientists to participate in the model training process.

To mitigate these human risks, pymetrics relies on a system of manual review that is similar to the code review practices that are common in serious software development projects. After a data scientist has finished training, evaluating, and packaging a model for given client they must complete a checklist that includes over 100 questions. These questions review all of the key aspects of the model building process: where did data come from, did it pass correctness checks, what types of game data were included, that the tree search was able to produce models that met minimum accuracy and fairness requirements, the pass rates for all demographic groups with respect to the "best" model, and various other model quality checks. The data scientist not only needs to check the boxes in the checklist, but also copy salient numerical data (e.g., accuracy and pass rates) into the sheet and document in writing any significant deviations from the standard "templated" model training process. While much of this process could, in theory, be automated, pymetrics has chosen deliberately to use a manual process that forces the data scientist to re-evaluate and justify all their work.

Before a trained model can be released into production, the corresponding notebook and checklist are reviewed by a second data scientist from pymetrics. The second data scientist provides extra bulwark against unintentionally erroneous models being accidentally put into production. Further, it would now take collusion between two data scientists to maliciously release an intentionally biased model into production.

In our opinion, this manual review process offers a reasonable level of assurance against malicious insiders and negligence.[9] The design of this review process forces self-reflection by data scientists and is so detail-oriented that it would be very difficult to unintentionally miss substantive problems with a trained model. The addition of a second reviewer guards against gross negligence and raises the bar against intentional malfeasance.

5.3 Cleaning and Imputation

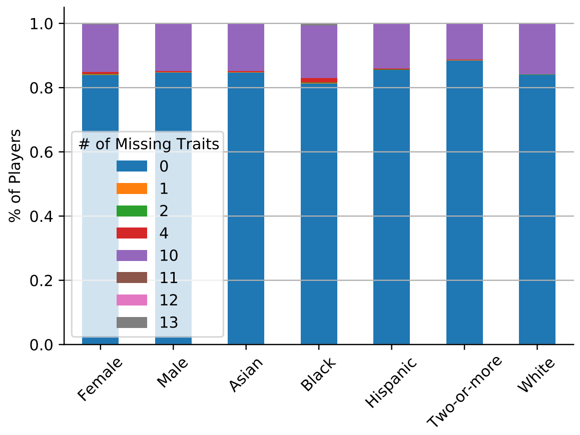
The next set of questions we investigate concern the **soundness of assumptions** underlying the process pymetrics uses to prepare data for model training and evaluation. Creating a data pipeline for ML tasks is never as simple as loading data and inputting it straight-away into a model training algorithm. Rather, as we described in § 3.2, data preparation is itself a complex task that involves many choices: what data to use, how to clean it to filter outliers, how to impute any missing values in the data, and how to normalize and scale the numerical data. These choices may impact model performance and fairness guarantees, so they are worth critically interrogating.

During this audit, we focused on imputation of missing values as the area of concern. We focus on imputation because it is not optional, it will impact players even if filtering is applied, and there is a

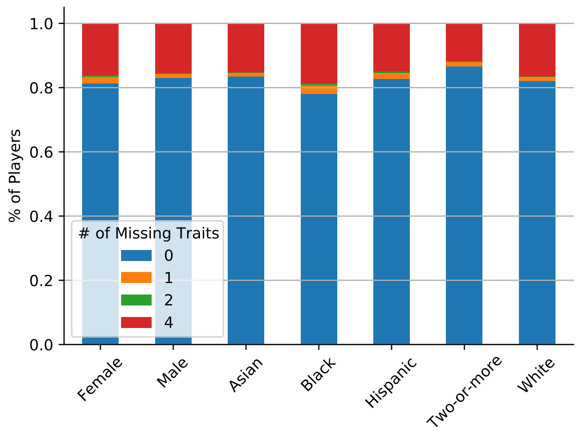
[9] We note that our opinion is based on assumptions about the threat model pymetrics faces, e.g., a malicious company that might want to get pymetrics to bless their biased hiring practices as fair. See § 5.1 for more discussion of this assumed threat model.

Missing	Demographic 1	Demographic 2	MW U	KW H
Games	Asian	Black	2864400.0	11.0*
	Black	White	2894400.0*	9.0*
Traits	Female	Male	14912935.0*	9.3*
	Asian	Black	2654601.0	37.0***
	Black	Hispanic	3043934.5***	19.0***
	Black	White	3069502.0***	26.0***
	Black	Two-or-More	1583887.0***	50.0***
	Hispanic	Two-or-More	1503170.0**	14.0**
	White	Two-or-More	1486928.5*	9.1*

Table 1: Cases where missing game and missing trait distributions are significantly correlated with demographics. $^*p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$.



(a) Balloons (13 total traits)



(b) Easy or Hard (7 total traits)

Figure 1: Distribution of missing traits for two games across demographic groups.

possibility that it may not impact all players equally (unlike scaling, which does effect all players equally).

5.3.1 The Differential Impact of Imputation

To motivate our study of imputation, we first perform statistical analyses to determine whether it may have differential impact on players with respect to their demographic groups. For this analysis we utilize the *bias group* dataset that we were given by pymetrics, since it includes players' demographic information.

First, we examine whether there are statistically significant differences between the distributions of missing games[10] and traits for different demographic groups. For each player in our dataset we count the number of missing traits and the number of missing games. Across the player population, we refer to these as the distribution of missing traits and games. We use the non-parametric Mann-Whitney U and Kruskal–Wallis H tests to identify missing trait and game distributions that are significantly different.[11] All p values are corrected for multiple hypothesis testing.

Table 1 shows the cases where we identified statistically significant differences in missing game and trait distributions. With respect to missing games, most demographics groups in our dataset do not show significant differences; Black players do show differences from Asian and White players but the significance level is relatively weak ($p < 0.05$). With respect to missing traits we find more cases of statistically significant differences. Male and female players have different distributions of missing traits ($p < 0.05$), Black players

[10]We borrow the same definition of "missing game" as pymetrics: for a given player, we define a game as missing if all of its associated traits are empty.

[11]Using the Anderson-Darling test, we found that almost none of the data we analyze in this section is normally distributed, thus we rely on non-parametric tests.

	count(MW U $p < 0.05$**)**	**%**	**count(KW** H $p < 0.05$**)**	**%**
Gender	19	29.7	30	46.9
Race/Ethnicity	112	17.5	214	33.4

Table 2: Count and percentage of cases where trait distributions are significantly different between demographic groups.

have different distributions than all other racial/ethnic groups ($p < 0.001$), and players with two-or-more groups have different distributions than most other groups (at varying significance levels.)

To illustrate these findings, Figure 1 shows examples of missing trait distributions for two pymetrics games. We see that, in general, ∼80% of players have no missing traits in these games, which is good: missing data is not the norm. However, amongst the remaining players, most are missing all or the majority of traits. Furthermore, we see that Black players in particular tend to be missing more traits.

Second, we examine whether there are statistically significant differences between the trait value distributions for different demographic groups. As before, we focus on data from the example *bias group* dataset, we use the non-parametric Mann-Whitney U and Kruskal–Wallis H tests to identify trait value distributions that are significantly different, and we correct p values for multiple hypothesis testing.

Table 2 shows the count (and percentage) of cases where two trait distributions were significantly different (at the $p < 0.05$ level), broken down by gender and race/ethnicity. Depending on the test metric, 30–47% of trait value distributions are significantly different along gender lines, while 18–33% are significantly different along race/ethnicity lines.

Based on these results, we draw two conclusions with respect to imputation. *First*, some demographic groups of players have more missing traits and therefore experience more imputation. This is particularly true for Black players. *Second*, demographic groups of players do exhibit different trait value distributions from the pymetrics games. Thus, setting missing trait values to the population median may misrepresent the gameplay characteristics of specific groups. Taken together, these two points motivate us to investigate the choice of imputation algorithm and its impact on trained models in greater detail.

5.3.2 Testing Imputation Algorithms

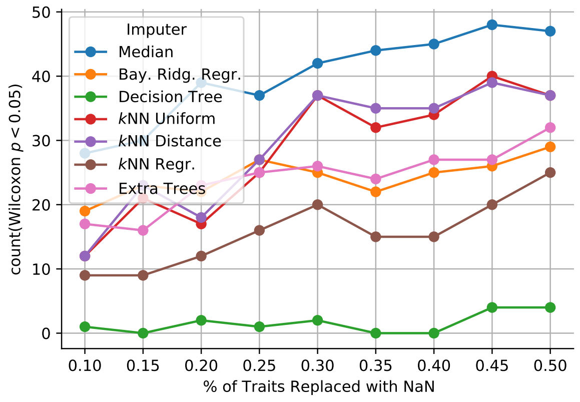
Now that we have established that there is potential for median imputation to have differential impact on players from different groups, we move on to evaluating different imputation algorithms. Our goal is to assess the degree to which a given imputation algorithm produces data that is statistically indistinguishable from ground-truth data, e.g., does the imputer do a good job at guessing reasonable values for missing traits?

We investigate seven imputation algorithms from the `scikit-learn` Python module version 0.23: median (the method currently used by pymetrics), Bayesian ridge regression, decision tree, extra trees (a.k.a. random forest), k nearest neighbors (kNN) with regression, kNN with uniform weighting, and kNN with distance-based weighting. We set the number of trees for extra trees to five, and the neighborhood size for the kNN algorithms to five.[12]

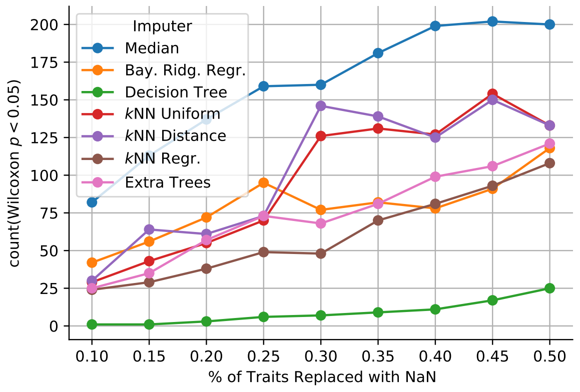
We follow the following procedure to test each imputation algorithm:

1. We filter out all players from the *bias group* dataset that have at least one missing trait. This reduces the size of the dataset from 10,786 to 8,120 players. We refer to this filtered dataset as *ground truth* because all of the trait values are known.
2. We randomly remove x% of trait values from each trait column. We use $x = [0.1, ..., 0.5]$.
3. We impute the missing values using a given imputation algorithm. We refer to this as an *imputed* dataset.
4. We compare the distributions of trait values from the *ground truth* dataset to the *imputed* dataset. For our metric we use the Wilcoxon signed-rank test because it is non-parametric and suitable for comparing related samples. We compare the overall distribution of values for each trait as well as the demographic-specific values for each trait.

[12]The AWS VM we were given by pymetrics did not have enough memory to support larger forests or cluster sizes.



(a) All trait values (max: 64 distributions)



(b) Trait values split by demographic groups (max: 448 distributions)

Figure 2: Evaluation of seven imputation algorithms. The x axis presents the fraction of ground-truth trait values that were replaced with *NaN*. The y axis is the count of trait distributions that showed statistically significant differences ($p < 0.05$, corrected for multiple tests) using the Wilcoxon signed-rank test when comparing the ground-truth to the imputed data.

5. After adjusting for multiple hypothesis testing, we count the cases where the result of the test is significant at $p < 0.05$, meaning that we can reject the null hypothesis that the trait distributions were drawn from the same distribution.

Figure 2 shows the results of our imputer tests. In each figure, the x-axis represents the percentage of trait values that we artificially nulled and then imputed, while the y-axis is the count of trait distributions that were significantly different at the $p < 0.05$ level. Lower numbers are better. Each line shows the performance of a different imputation algorithm, and the subfigures break down the results by overall trait distributions and demographically-stratified trait distributions.

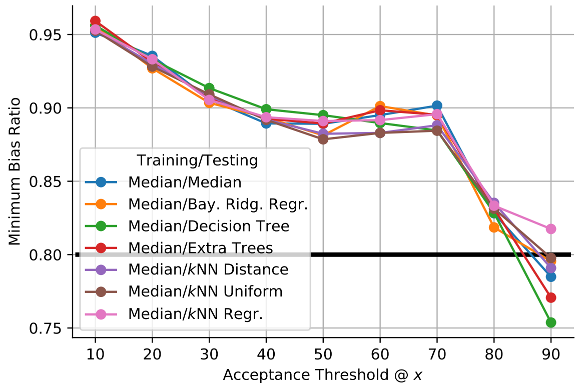
The results across Figure 2a and Figure 2b are consistent: the decision tree imputer produces trait distributions that are very difficult to distinguish from ground-truth, even when the distributions are subdivided by demographic. kNN with regression, Bayesian ridge regression, and extra trees come in second, third, and fourth place, respectively, while median performs worst.

5.3.3 Adverse Impact Testing and Model Performance

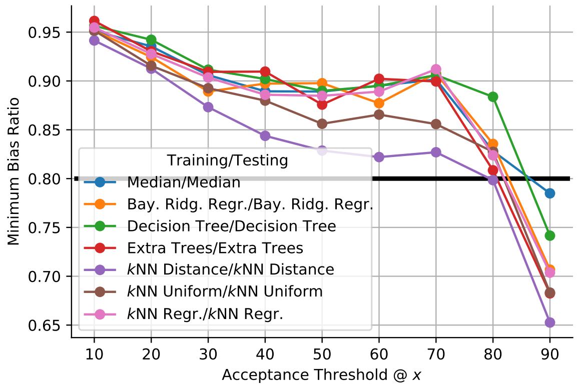
The results in the previous section demonstrate that not all imputation algorithms exhibit equally good performance on our *bias group* dataset. This motivates our final series of tests: training and evaluating models using the different imputation algorithms to determine if the choice of imputer has a substantive impact on the fairness guarantees and performance of trained models.

First, we re-examine the fairness guarantees offered by pymetrics "default" model. Recall that pymetrics trains models using a dataset that uses median imputation, and then performs adverse impact testing using a dataset that is also median imputed. As we we showed in § 5.3.2, however, the median imputer produces datasets that may not reflect ground-truth. This raises the spectre that relying on median imputed data for adverse impact testing may generate misleading results, e.g., a belief that a model obeys the four-fifths rule when it does not, given higher-quality data.

To test this hypothesis, we re-evaluated the minimum bias ratios for the model that pymetrics' data scientists selected as best for our *in group*, *out group*, and *bias group* datasets. Figure 3a shows the results when we re-evaluate the original model (which we denote as "Median" since it was trained on a median imputed dataset) using *bias group* datasets that were imputed with varying algorithms. "Median/Median" presents the original fairness metrics computed by pymetrics. The x-axis denotes the acceptance threshold for employment candidates, with higher numbers corresponding to more stringent thresholds. The y-axis



(a) Models trained using a dataset with median imputation, but adverse impact tested using a datasets imputed with varying algorithms.



(b) Models trained and adverse impact tested using datasets imputed with varying algorithms.

Figure 3: Minimum bias ratio at different acceptance thresholds for models trained and tested using varying combinations of data imputed using different imputation algorithms.

Training/Testing	**# Features**	**AUC**	**Accuracy**	**Precision**	**Recall**
Median/Median	44	0.72	0.72	0.62	0.75
Bay. Ridg. Regr./Bay. Ridg. Regr.	45	0.72	0.71	0.61	0.76
Decision Tree/Decision Tree	45	0.71	0.71	0.63	0.68
Extra Trees/Extra Trees	45	0.72	0.72	0.62	0.77
kNN Distance/kNN Distance	56	0.70	0.70	0.61	0.70
kNN Uniform/kNN Uniform	53	0.70	0.69	0.60	0.71
kNN Regr./kNN Regr.	44	0.70	0.69	0.59	0.75

Table 3: Summary information for models trained and tested using datasets imputed via different algorithms. For a given imputer, we chose the model whose performance characteristics most closely resembled the original "Median/Median" model chosen by pymetrics data scientists.

denotes the minimum bias ratio for the given model/evaluation data at a given acceptance threshold.

Figure 3a demonstrates that the choice of imputation algorithm does not substantively alter the adverse impact assessment of the original pymetrics model. Despite using "better" evaluation datasets (i.e., produced by more accurate imputation algorithms), this model still passes the four-fifths test up to acceptance thresholds of 80. We also observe little variability at acceptance threshold less than 90, indicating that the fairness assessment of the model is stable.

Second, we ask a different question: can we train fairer models if we rely on datasets that were imputed using more accurate algorithms? To answer this question, we trained new models using *out group* datasets that were imputed using a variety of algorithms and evaluate them for adverse impact using *bias group* datasets imputed using corresponding algorithms. Recall that pymetrics uses a tree search to train many candidate models; for each imputed *out group* dataset we executed the tree search and then selected the model with the closest performance (AUC, accuracy, precision, and recall) to the original, median imputed model.

Figure 3b shows how minimum bias ratio varies with acceptance threshold for our newly trained models (and the original "Median" pymetrics model). In this case we do observe more variability in the performance of the models. The model trained on decision tree imputed data is able to achieve a significantly higher minimum bias ratio at the 80 threshold, making it the "fairest" model overall. In contrast, the kNN uniform and kNN distance imputers produce models with the worst fairness.

Table 3 shows the performance characteristics of models trained using *out group* datasets with varying imputation algorithms. Although we tried to select models with the same or better performance than the original "Median" imputed model, our results are mixed. The "Extra Trees" model slightly outperforms the original model (in recall), and the "Bayesian Ridge Regression" model achieves rough performance parity. In contrast, the "Decision Tree" model has much lower recall (0.68 versus 0.75 in the original model), which is unfortunate because as we show in Figure 3b it is the fairest model.

Taken together, the results in Figure 3b and Table 3 suggest that the extra trees imputation algorithm may be the best choice, since it offers similar minimum bias ratio results as the median imputer and slightly greater recall. However, we caution that these findings may not generalize to other datasets — additional testing is necessary with other *in group*, *out group*, and *bias group* datasets.

Furthermore, based on the analysis in this section, we conclude that the median imputer used by pymetrics does produce models that have sound fairness guarantees. Our testing demonstrates that the issues we uncovered with median imputation in § 5.3.2 do not have a significant impact on the calculation of minimum bias ratios or compliance with the four-fifths rule.

6 Conclusion

In this study, we present the process by which we audited pymetrics' candidate screening tool and the results of our audit. The focus of our audit was on pymetrics' claim that their trained ML models conform to the UGESP four-fifths rule using the minimum bias ratio metric. We conducted our audit in summer 2020 based on documentation, source code, and representative datasets that pymetrics provided to us.

With respect to process, we present a novel set of procedures and safeguards for conducting cooperative algorithm audits between public and private entities. To the best of our knowledge, our audit of pymetrics may be the first publicly-documented algorithm audit between a willing private company and an external investigative team. As such, we had to navigate challenging questions around how to structure our audit with respect to scoping our research questions, accepting remuneration, maintaining the security of confidential source code and data, and preserving investigative secrecy, while simultaneously maintaining an arms-length, objective relationship with pymetrics. We hope that this audit sets a new precedent for cooperative algorithm audits, and that this leads to more companies engaging independent experts to audit their sociotechnical systems in the future.

6.1 Results

With respect to the results of our audit, we are comfortable stating that pymetrics passed the audit, subject to the qualifications and limitations we state in § 4.2 and § 4.4. Based on extensive examination and testing of their source code, we can confirm that pymetrics' claims about the process they use to conduct adverse impact testing of models is implemented faithfully in their source code, and that it does result in models that conform to the four-fifths rule with respect to gender and ethnicity/race (as defined by the corresponding EEOC categories). We were unable to bypass their adverse impact testing procedure by crafting malicious datasets, and we judge their human compliance safeguards as sufficient to offer reasonable assurance that human error is unlikely to result in an unfair model being accidentally deployed.

6.2 Representativeness of the Bias Group

One aspect of pymetrics that we were unable to audit concerns the *bias group* dataset that pymetrics uses to perform adverse impact testing. Although the *bias group* for a given engagement is randomly selected from a large population (600,000+ people, according to pymetrics [40]) of historical game players, this does not necessarily mean that it is representative of potential job seekers. Since the *bias group* serves as the baseline for adverse impact testing, if it is not representative then the resulting bias assessment may be flawed. We identify two issues:

1. Only players who opt-in to a demographic survey are eligible for inclusion in the *bias group*. Although pymetrics claims that the overall response rate to the survey is high (over 75% [40])), that does not necessarily mean that the response rate is independent of demographics.
2. The *bias group* is drawn from the gameplay data of people who have played pymetrics' games in the past, which is itself conditioned on the types of companies, job roles, and job locations that pymetrics has solicited gameplay data for in the past.

Pymetrics identified the first issue themselves and performed an internal study to investigate the issue on 2019. pymetrics provided a copy of this study to us, updated to reflect the growth of their dataset as of summer 2020 [40]. The majority of the study focuses on a dataset drawn from four clients that agreed to disclose the demographic data of job applicants — pymetrics was then able to compare the data collected by the clients to the demographic data pymetrics collected during the corresponding gameplay sessions. In total this dataset covers around 40,000 people, with roughly 5,600 people confirmed to appear in both the client provided and pymetrics datasets.

Pymetrics' study found that there was effectively no relationship between demographics and disclosure rates. Overall, pymetrics observed that (1) people where more likely to reveal their demographics to pymetrics than to clients, (2) that this behavior was consistent across clients and demographic groups, and most importantly (3) the proportion of people who revealed their demographics was consistent across the pymetrics and client datasets. Taken together, these results provide some reassurance that biased survey response rates are not undermining the *bias group*. That said, the only way to revolve this issue definitely would be through a deep ethnographic study of job seekers.

With respect to the second issue, pymetrics provided us with two datasets to shed light on the diversity of the *bias group* dataset. *First*, they provided high-level summary data on the geographic distribution of players in the *bias group*, aggregated by country. pymetrics has received data from players in 191 countries, with ∼40% coming from the US, but also with significant numbers around North and South America, Europe, Southeast and East Asia, South Africa, and Australia. *Second*, they provided a dataset that mapped their 600+ active client engagements from January to October 2020 to O*NET occupations.[13] In terms of coverage, as of this writing pymetrics had developed models for jobs that cover 16 of the 23 major O*NET groups (70% coverage) and 35 of the 98 minor groups (36%). We do not expect pymetrics to cover all of these groups, such as Group 45: Farming, Fishing, and Forestry Occupations and Group 55: Military Specific Operations, since employers in these groups are unlikely to rely on predictive analytics for recruitment.

As above, these datasets provide some reassurance that pymetrics *bias group* dataset is relatively diverse along geographic and job category lines, although there is no guarantee that it is sufficiently diverse to cover all potential recruiting use cases.

Ultimately, our concerns about the representativeness of the *bias group*, and its potential impact on models' fairness guarantees, are only valid up to a point. As we note in § 3, pymetrics performs post-hoc adverse impact testing on models after they are deployed, based on the data of players who specifically applied to the corresponding jobs. If these tests reveal that a model is not living up to the promised fairness guarantees, then pymetrics decommissions the model and trains a replacement that is fair using the updated data. Given that there are an enormous number of factors that might cause the applicant pool for a particular job to diverge from a given reference population used for pre-deployment adverse impact testing, post-hoc testing is a reasonable mitigation for identifying instances where modeling and testing assumptions diverge from reality.

6.3 Future Work

The context surrounding pymetrics afforded us the privilege of narrowly scoping our audit. Unlike algorithms in many contexts, hiring and employment are a rare area that is strictly regulated, with clear compliance metrics that are relatively straightforward to operationalize, and that are supported by a long history of jurisprudence [31, 48, 1]. That said, there two ways that we could have expanded our audit, and that hiring companies in the future could be audited.

Our first open question concerns pymetrics' choice to focus exclusively on disparate impact (and disparate treatment) in their fairness testing. The choice to operationalize the four-fifths rule potentially privileges regulatory compliance above the values of all other stakeholders, e.g., job seekers who may care more about differential validity [48]. Pymetrics does offer to customize their adverse impact testing to suite clients' needs, but this is not the same as fundamentally re-evaluating the default codebase and the guarantees it offers. It remains to be seen whether multiple fairness guarantees can be met in the context of pymetrics business while still preserving pymetrics' conservative stance with respect to regulatory compliance.

[13]O*NET is a hierarchical taxonomy of employment areas, broad occupations, and detailed occupations developed and maintained by the U.S. Office of Management and Budget and the U.S. Department of Labor.

Our second open question that could be audited is the efficacy of pymetrics' games at assessing the "fit" of job seekers. Pymetrics' games are based on peer-reviewed and replicated psychological studies, but drawing a direct line from laboratory experiments to real-world job performance is challenging. Pymetrics provided us with a confidential presentation containing results from game validation studies [4] — at a minimum we encourage pymetrics to make these results public. That said, larger-scale, observational studies based on the longitudinal data pymetrics collects from clients to evaluate model performance could be invaluable for assessing the efficacy of the games.

References

[1] Ifeoma Ajunwa. The paradox of automation as anti-bias intervention. *Cardozo, L. Rev.*, 41, 2020.

[2] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias. ProPublica, 2016. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

[3] Joshua Asplund, Motahhare Eslami, Hari Sundaram, Christian Sandvig, and Karrie Karahalios. Auditing race and gender discrimination in online housing markets. In *Proc of ICWSM*, 2020.

[4] Lewis Baker. [Confidential] Games, Measures and Factors: Measurement Validity. pymetrics, inc., 2019.

[5] Jack Bandy and Nicholas Diakopoulos. Auditing news curation systems: A case study examining algorithmic and editorial logic in apple news. In *Proc of ICWSM*, 2020.

[6] Solon Barocas and Andrew D. Selbst. Big data's disparate impact. *104 California Law Review*, 671, 2016.

[7] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Proc. of FAT**, 2018.

[8] Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186, April 2017.

[9] Le Chen, Ruijun Ma, Anikó Hannák, and Christo Wilson. Investigating the impact of gender on rank in resume search engines. In *Proc. of CHI*, 2018.

[10] Le Chen, Alan Mislove, and Christo Wilson. Peeking Beneath the Hood of Uber. In *Proc. of IMC*, October 2015.

[11] Le Chen, Alan Mislove, and Christo Wilson. An Empirical Analysis of Algorithmic Pricing on Amazon Marketplace. In *Proc. of WWW*, 2016.

[12] Alexandra Chouldechova, Diana Benavides-Prado, Oleksandr Fialko, and Rhema Vaithianathan. A case study of algorithm-assisted decision making in child maltreatment hotline screening decisions. In *Proc. of FAT**, 2018.

[13] Nicholas Diakopoulos. Algorithmic accountability reporting: on the investigation of black boxes. Tow Center for Digital Journalism Brief, 2014.

[14] Nicholas Diakopoulos, Daniel Trielli, Jennifer Stark, and Sean Mussenden. I Vote For—How Search Informs Our Choice of Candidate. In M. Moore and D. Tambini, editors, *Digital Dominance: The Power of Google, Amazon, Facebook, and Apple*, page 22. 2018.

[15] Danielle Ensign, Sorelle A. Friedler, Scott Neville, Carlos Scheidegger, and Suresh Venkatasubramanian. Runaway feedback loops in predictive policing. In *Proc. of FAT**, 2018.

[16] Equal Employment Opportunity Commission, Civil Service Commission, et al. Uniform guidelines on employee selection procedures. *Federal Register*, 43(166):38290–38315, 1978.

[17] Motahhare Eslami, Kristen Vaccaro, Karrie Karahalios, and Kevin Hamilton. "be careful; things can be worse than they appear": Understanding biased algorithms and users' behavior around them in rating platforms. In *Proc of ICWSM*, 2017.

[18] Motahhare Eslami, Kristen Vaccaro, Min Kyung Lee, Amit Elazari Bar On, Eric Gilbert, and Karrie Karahalios. User attitudes towards algorithmic opacity and transparency in online reviewing platforms. In *Proc. of CHI*, 2019.

[19] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proc. of KDD*, 2015.

[20] Sorelle A. Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. On the (im)possibility of fairness. *CoRR*, abs/1609.07236, 2016.

[21] Sorelle A. Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P. Hamilton, and Derek Roth. A comparative study of fairness-enhancing interventions in machine learning. In *Proc. of FAT**, 2019.

[22] Batya Friedman and Helen Nissenbaum. Bias in computer systems. *ACM Trans. Inf. Syst.*, 14(3):330–347, July 1996.

[23] Aniko Hannak, Piotr Sapieżyński, Arash Molavi Kakhki, Balachander Krishnamurthy, David Lazer, Alan Mislove, and Christo Wilson. Measuring Personalization of Web Search. In *Proc. of WWW*, 2013.

[24] Aniko Hannak, Gary Soeller, David Lazer, Alan Mislove, and Christo Wilson. Measuring price discrimination and steering on e-commerce web sites. In *Proc. of IMC*, 2014.

[25] Anikó Hannák, Claudia Wagner, David Garcia, Alan Mislove, Markus Strohmaier, and Christo Wilson. Bias in online freelance marketplaces: Evidence from taskrabbit and fiverr. In *Proc of CSCW*, 2017.

[26] Desheng Hu, Shan Jiang, Ronald E. Robertson, and Christo Wilson. Auditing the partisanship of google search snippets. In *Proc. of WWW*, 2019.

[27] Eslam Hussein, Prerna Juneja, and Tanushree Mitra. Measuring misinformation in video search platforms: An audit study on youtube. *Proc. ACM Hum.-Comput. Interact.*, 4(CSCW1), May 2020.

[28] Shan Jiang, Le Chen, Alan Mislove, and Christo Wilson. On ridesharing competition and accessibility: Evidence from uber, lyft, and taxi. In *Proc. of WWW*, 2018.

[29] Anna Kawakami, Khonzoda Umarova, Dongchen Huang, and Eni Mustafaraj. The 'fairness doctrine' lives on? theorizing about the algorithmic news curation of google's top stories. In *Proc. of HT*, 2020.

[30] Matthew Kay, Cynthia Matuszek, and Sean A. Munson. Unequal representation and gender stereotypes in image search results for occupations. In *Proc. of CHI*, 2015.

[31] Pauline T. Kim. Data-driven discrimination at work. *William & Mary Law Review*, 58, 2017.

[32] Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. *CoRR*, abs/1609.05807, 2016.

[33] Chloe Kliman-Silver, Aniko Hannak, David Lazer, Christo Wilson, and Alan Mislove. Location, Location, Location: The Impact of Geolocation on Web Search Personalization. In *Proc. of IMC*, 2015.

[34] Juhi Kulshrestha, Motahhare Eslami, Johnnatan Messias, Muhammad Bilal Zafar, Saptarshi Ghosh, Krishna P. Gummadi, and Karrie Karahalios. Quantifying search bias: Investigating sources of bias for political searches in social media. In *Proc of CSCW*, 2017.

[35] Peter Lee. Learning from Tay's Introduction. Official Microsoft Blog, 2016. https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction/.

[36] Zachary Lipton, Julian McAuley, and Alexandra Chouldechova. Does mitigating ML's impact disparity require treatment disparity? In *Proc. of NeurIPS*, 2018.

[37] Kristian Lum and William Isaac. To predict and serve? *Significance*, 13(5), 2016.

[38] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proc. of NIPS*, 2017.

[39] Emma Lurie and Eni Mustafaraj. Investigating the effects of google's search engine result page in evaluating the credibility of online news sources. In *Proc. of WebSci*, 2018.

[40] Matthew Mol. [Confidential] Demographic Disclosure Study. pymetrics, inc., 2020.

[41] Dino Pedreshi, Salvatore Ruggieri, and Franco Turini. Discrimination-aware Data Mining. In *Proc. of KDD*, 2008.

[42] Brad Plumer and Nadja Popovich. How decades of racist housing policy left neighborhoods sweltering. The New York Times, 2020. https://www.nytimes.com/interactive/2020/08/24/climate/racism-redlining-cities-global-warming.html.

[43] pymetrics. [Confidential] Fairness Testing Procedures, 2019.

[44] pymetrics. [Confidential] Job Analysis Methods & Process, 2019.

[45] pymetrics. [Confidential] Technical Brief for pymetrics, 2019.

[46] pymetrics Data Science Team. Open Sourced Bias Testing for Generalized Machine Learning Applications (audit-AI). Github, 2020. https://github.com/pymetrics/audit-ai.

[47] Lincoln Quillian, Devah Pager, Ole Hexel, and Arnfinn H. Midtbøen. Meta-analysis of field experiments shows no change in racial discrimination in hiring over time. *Proceedings of the National Academy of Sciences*, 114(41):10870–10875, 2017.

[48] Manish Raghavan, Solon Barocas, Jon Kleinberg, and Karen Levy. Mitigating bias in algorithmic hiring: Evaluating claims and practices. In *Proc. of FAT**, 2020.

[49] Inioluwa Deborah Raji, Andrew Smart, Rebecca N. White, Margaret Mitchell, Timnit Gebru, Ben Hutchinson, Jamila Smith-Loud, Daniel Theron, and Parker Barnes. Closing the ai accountability gap: Defining an end-to-end framework for internal algorithmic auditing. In *Proc. of FAT**, 2020.

[50] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proc. of KDD*, 2016.

[51] Ronald E Robertson, Shan Jiang, Kenneth Joseph, Lisa Friedland, David Lazer, and Christo Wilson. Auditing partisan audience bias within google search. *Proceedings of the ACM: Human-Computer Interaction*, 2(CSCW), November 2018.

[52] Ronald E. Robertson, Shan Jiang, David Lazer, and Christo Wilson. Auditing autocomplete: Recursive algorithm interrogation and suggestion networks. In *Proc. of WebSci*, 2019.

[53] Ronald E Robertson, David Lazer, and Christo Wilson. Auditing the personalization and composition of politically-related search engine results pages. In *Proc. of WWW*, 2018.

[54] Christian Sandvig, Kevin Hamilton, Karrie Karahalios, and Cedric Langbort. Auditing algorithms: Research methods for detecting discrimination on internet platforms. In *In Prov. of Data and Discrimination: Converting Critical Concerns into Productive Inquiry, a preconference at the Annual Meeting of the International Communication Association*, 2014.

[55] Gary Soeller, Karrie Karahalios, Christian Sandvig, and Christo Wilson. MapWatch: Detecting and Monitoring International Border Personalization on Online Maps. In *Proc. of WWW*, 2016.

[56] Latanya Sweeney. Discrimination in Online Ad Delivery. *ACM Queue*, 11(3), April 2013.

[57] U.S. Congress. Civil Rights Act, 1964.

[58] Giridhari Venkatadri, Yabing Liu, Athanasios Andreou, Oana Goga, Patrick Loiseau, Alan Mislove, and Krishna P. Gummadi. Privacy Risks with Facebook's PII-based Targeting: Auditing a Data Broker's Advertising Interface. In *Proc. of IEEE Symposium on Security and Privacy*, 2018.

[59] Giridhari Venkatadri, Elena Lucherini, Piotr Sapiezyński, and Alan Mislove. Investigating sources of PII used in Facebook's targeted advertising. In *Proc. of PETS*, Jul 2019.

[60] James Vincent. These stickers make computer vision software hallucinate things that aren't there. The Verge, 2018. https://www.theverge.com/2018/1/3/16844842/ai-computer-vision-trick-adversarial-patches-google.

[61] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P. Gummadi. Fairness Constraints: Mechanisms for Fair Classification. In *Proc. of International Conference on Artificial Intelligence and Statistics*, 2017.