# Data Forging Attacks on Cryptographic Model Certification

**Carter Luck**
University of Massachusetts Amherst

**Olive Franzese**
Vector Institute & University of Toronto

**Elisaweta Masserova**
Carnegie Mellon University

**Akira Takahashi**
J.P.Morgan AI Research & AlgoCRYPT CoE

**Antigoni Polychroniadou**
J.P.Morgan AI Research & AlgoCRYPT CoE

## Abstract

Privacy-preserving machine learning auditing protocols allow auditors to assess models for properties such as fairness or robustness, without revealing their internals or training data. This makes them especially attractive for auditing models deployed in sensitive domains such as healthcare or finance. For these protocols to be truly useful, though, their guarantees must reflect how the model will behave once deployed, not just under the conditions of an audit. Existing security definitions often miss this mark: most certify model behavior only on a *fixed audit dataset*, without ensuring that the same guarantees *generalize* to other datasets drawn from the same distribution. We show that a model provider can attack many cryptographic model certification schemes by forging training data, resulting in a model that exhibits benign behavior during an audit, but pathological behavior in practice. For example, we empirically demonstrate that an attacker can train a model that achieves over 99% accuracy on an audit dataset, but less than 30% accuracy on fresh samples from the same distribution.

## 1 Introduction

Certifiable, privacy-preserving machine learning aims to formally prove desired properties of the model while keeping model parameters and training data confidential [Zhang et al., 2020, Liu et al., 2021, Shamsabadi et al., 2022]. In this context, the typical lifecycle follows a sequence in which the model provider first trains the model, then an auditor evaluates it according to desired criteria, and—after passing the audit —the certified model is deployed.[1] Note that certification comes from the use of cryptography (e.g., cryptographic commitments, zero-knowledge proofs [Goldwasser et al., 1985]) rather than a specific ML algorithm. The usage of cryptographic techniques allows to not only certify the intended property, but do so while keeping the model internals and training data private. However, it turns out that the guarantees that model certifications provide are bound to the specific dataset that was used during the audit (e.g., "a demographic parity gap of the model held by the provider is below 10% on the UCI Default Credit dataset"). In this paper, we observe that such dataset-specific guarantees risk creating a false sense of security: by themselves, they do not ensure that the certified properties will continue to hold once the model is deployed and applied

[1] Some works require continuous auditing during deployment instead of a single audit pre-deployment; see Table 1.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Regulatable Machine Learning.

Table 1: Analysis of vulnerabilities to data-forging attacks in privacy-preserving ML audits. ✓= supported; ▲= conditional; ✗= not supported.

| Work | **Certified property** | | | | **Resilience to data-forging** | **Continuous verification** |
|---|---|---|---|---|---|---|
| | Acc. | Group Fair | Indv. Fair | Diff. Priv. | | |
| Zhang et al. [2020] | ✓ | ✗ | ✗ | ✗ | ▲ (pd) | ✗ |
| Shamsabadi et al. [2022] | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ |
| Yadav et al. [2024] | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ |
| Liu et al. [2021] | ✓ | ✗ | ✗ | ✗ | ▲ (pd) | ✗ |
| Franzese et al. [2024] | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ |
| Shamsabadi et al. [2024] | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ |
| Kang et al. [2022] | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ |
| Wang and Hoang [2023] | ✓ | ✗ | ✗ | ✗ | ▲ (pd) | ✗ |
| Bourrée et al. [2025] | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ |

Acc. = accuracy; Group/Indv. Fair = group/individual fairness; Diff. Priv.=differential privacy. "Conditional" works lack detail to assess resilience to data-forging, but indicate deployments with public datasets (pd), which would be make the solution vulnerable. Continuous verification means audits must run continuously during deployment (e.g., via clients) rather than once pre-deployment.

to *fresh data*, even when this data is drawn from the same distribution as the audit dataset. We show that this is not merely a theoretical concern.

We propose novel attack strategies allowing an adversarial model provider to pass an audit (thus enabling deployment) while simultaneously pursuing its own, potentially conflicting, interests. For example, while an auditor may seek to verify fairness, the model owner may instead prioritize accuracy—even when accuracy and fairness are in tension. We show that when the audit dataset is known in advance (as is often the case when public benchmark datasets are used), the model owner can carefully engineer "training data" so that a model honestly trained on it passes the audit, while exhibiting pathological behavior on real-world inputs. We empirically show that such **data forging attacks** can cause dramatic gaps between audit-time guarantees and true model performance: for instance, in one of our attacks a model can pass an audit requiring 80% accuracy on the audit dataset, yet achieve only 30% accuracy on new samples from the same distribution. We establish the attacks rigorously for decision trees—both empirically and formally—and provide preliminary empirical evidence for neural networks. We show that our attacks remain undetected by straightforward approaches such as statistical tests, e.g., Welch's $t$-test [Welch, 1947] are performed to check whether the training data and audit data were taken from the same distribution. We further show that a number of prior works are vulnerable to such data forging attacks (see Table 1).

Motivated by these vulnerabilities, we introduce a unified syntax capturing existing auditing schemes and define a formal attack game that highlights the gap between certifying a property on a fixed dataset and certifying that the property generalizes to fresh samples from the distribution. Given the existence of data-forging attacks in this setting, we underscore the importance of conducting audits on test data that is *independent* of both the model and its training data. This approach may serve as a template for future work to achieve auditing solutions that are not only efficient but also secure.

In summary, our work advances the study of cryptographic auditing for machine learning by **(i)** proposing a novel attack strategy that passes an audit while enabling pathological model behavior at deployment with respect to real-world inputs, and **(ii)** empirically demonstrating the effectiveness of our attack against three example certification objectives: accuracy auditing, fairness auditing, and statistics for distribution similarity testing. We emphasize that we *do not suggest that prior cryptographic works are broken on a technical level*, rather that the guarantees these works provide deserve closer scrutiny. Our findings comprise strong evidence that secure audit solutions with any of the following properties are unlikely: a) those which utilize known public datasets for test purposes, and b) those that reuse test datasets (if model owner learns a substantial amount of this test dataset during the audit). This evidences the importance of continuous sampling of fresh data for a successful audit infrastructure. We hope that our work will inform the design of future cryptographically secure machine learning audit frameworks.

## 2 Related Work

Our work is related to, but distinct from, data poisoning attacks [Steinhardt et al., 2017]. Such attacks have traditionally been considered in the context of machine learning systems trained on user-provided data. Both data poisoning attacks and the concrete attacks in our work (see §4.1) involve adversarial manipulations of training data. However, the data poisoning setting is conceptually different from ours: In data poisoning, the model provider is typically considered honest, and the concern is that users contributing to the model can inject malicious data to degrade a model's performance. As a result, data poisoning involves subtle, often small-scale perturbations to the training data. More formally, data poisoning can be viewed as a game between a *defender*, who seeks to learn an accurate model, and an *attacker*, who wishes to corrupt the learned model [Barreno et al., 2010]. The model is honestly trained on the combination of a clean dataset $D_c$ and a poisoned dataset $D_p$, where the size of $D_p$ is no larger than that of $D_c$. In contrast, we consider a fully malicious model provider. Its goal is to engineer a model that passes an audit, while violating the certified properties on real-world data. Our adversary is not restricted to small-scale perturbations of the clean training data and is not required to perform the training in an honest way.

The conclusions we draw about requiring fresh data for auditing are semantically related to work on the inadequacy of public benchmarks in machine learning Zhang et al. [2025a], Hardt [2025], but those works do not consider cryptographic security. For additional related work and an overview of certifiable ML, see §E.

## 3 Certifying ML: Background and Unifying Syntax

Consider the following scenario: An auditor wishes to verify whether a model utilized by an insurance company to justify claim decisions (approve/deny claim) is accurate on a dataset of the auditor's choosing. At the same time, the company does not want to reveal its model due to concerns about privacy and business competition. Certifiable ML works use cryptographic techniques to reconcile these seemingly conflicting goals.

**Zero-knowledge proofs** Among these techniques, the central tool is *zero-knowledge (ZK) proofs*, a classical cryptographic primitive, which allows one party (a *prover*) prove a statement $\mathsf{x}$ to another party (*verifier*) without revealing anything else apart from the validity of this statement. Such proofs are constructed for a concrete NP relation $\mathcal{R}$, which is used to formalize what it means for a statement to be true by specifying the type of evidence (witness $\mathsf{w}$) that certifies it. The statement $\mathsf{x}$ is public, the witness $\mathsf{w}$ is private, and the zk proof checks $(\mathsf{x}, \mathsf{w}) \in \mathcal{R}$, without revealing $\mathsf{w}$. In certifiable ML, such proofs allow model provider (prover) to formally prove that a model (witness) satisfies a desired property (e.g., accuracy, fairness, or inference correctness) on a given test dataset (statement) without learning anything else about the model or the training data. More formally:

**Definition 1** (Proof System)**.** *An (interactive)* proof system $\mathsf{ZKP}$ *for an NP relation $\mathcal{R}$ is a tuple of interactive Turing machines $(\mathcal{P}, \mathcal{V})$, where $\mathcal{P}$ is prover and $\mathcal{V}$ is verifier. Let $b \leftarrow \langle \mathcal{P}(\mathsf{w}), \mathcal{V} \rangle(\mathsf{x})$ denote the interaction between $\mathcal{P}$ and $\mathcal{V}$, where both $\mathcal{P}$ and $\mathcal{V}$ take $\mathsf{x}$ as common inputs, and $\mathcal{P}$ additionally takes $\mathsf{w}$ as a private input. At the end of interaction, $\mathcal{V}$ halts by outputting a binary $b$.*

Proof systems that are used in ML auditing typically require the following security properties: For an NP relation $\mathcal{R}$, they must provide *completeness* (i.e., if prover and verifier follow the protocol with input $(\mathsf{x}, \mathsf{w}) \in \mathcal{R}$, verifier always accepts), *(knowledge) soundness* (i.e., if verifier accepts, then it must be that prover owns a valid witness $\mathsf{w}$ satisfying given NP relation w.r.t. statement $\mathsf{x}$), and *zero knowledge* (i.e., the transcript of the interaction between the prover and the (malicious) verifier leaks nothing except that there exists a witness $\mathsf{w}$ such that $(\mathsf{x}, \mathsf{w}) \in \mathcal{R}$). See §A.4 for formal definitions and §F for an overview of the NP relations underlying common zk proofs in certifiable ML (e.g., proofs of training, inference, etc.).

Returning to our example, suppose the insurance company has successfully passed an audit and can now deploy its model. How can a customer submitting inference queries be assured that the company continues to use the *certified* model—rather than switching to a different, unverified one? Again, the company still wishes to keep its model private.

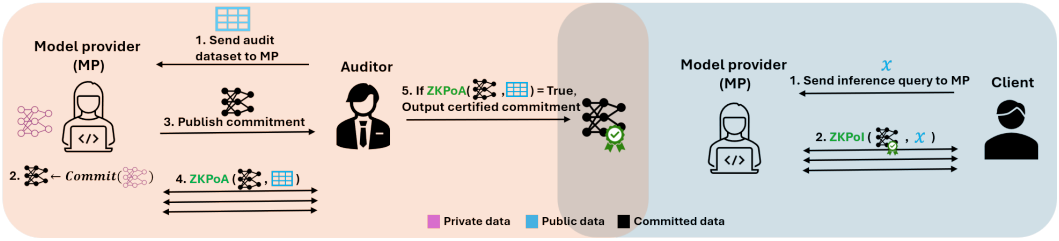


Figure 1: Simplified protocol flow for (insecure) ZK-based ML certification. Left: The model provider, after observing the audit dataset, commits to a model and engages with the auditor in a zero-knowledge proof of accuracy (ZKPoA). If the audit succeeds, the auditor certifies the committed model. Right: For each new inference query, the model provider interacts with the client in a zero-knowledge proof of inference (ZKPoI) protocol, ensuring that the result is consistent with the previously certified commitment.

**Cryptographic Commitment Schemes** The standard cryptographic tool here is *commitments*, which bind the provider to a single private model during the audit. This prevents "model switching" and ensures that model used in deployment is the same as the one that was certified:

**Definition 2** (Commitment Scheme)**.** *A commitment scheme is an algorithm* Commit*, which is executed as* $\mathsf{com} \leftarrow \mathsf{Commit}(m; \rho)$*. It takes as input a message* $m \in \{0,1\}^{\ell_m(\lambda)}$*, a uniformly sampled randomness* $\rho \in \{0,1\}^{\ell_r(\lambda)}$*, and returns a commitment* $\mathsf{com} \in \{0,1\}^{\ell_c(\lambda)}$*. Here* $\ell_m, \ell_r, \ell_c$ *are some polynomials in* $\lambda$*, the security parameter (determining the desired level of security).*

We require two security properties: *hiding* (i.e., given a commitment com, it leaks nothing about the message $m$), and *binding* (i.e., it is computationally infeasible to find two different pairs $(m, \rho)$ and $(m', \rho')$ such that $\mathsf{Commit}(m; \rho) = \mathsf{Commit}(m'; \rho')$). See §A.5 for formal definitions.

Now, auditing may require publishing such a commitment to the model,[2] after which the client and insurance company engage in a ZK proof of inference against it. Figure 1 shows the full certification workflow, where the audit dataset is revealed to the model provider prior to committing to the model.

**Unifying Syntax for Prior Works** We will next discuss the security guarantees of works that address the first stage of certification—namely, proofs of accuracy, fairness, etc., between auditor and model provider. To analyze these systematically, rather than case by case, we abstract away implementation details and introduce a unifying syntax that captures a broad class of existing audit systems.

Given a predicate $f(h, S_{train}, S_{audit})$ and a distribution $\mathcal{D}$, we define the auditing scheme as follows:

1. Auditor samples $S_{\text{audit}} \sim \mathcal{D}$ (or uses a public one) and sends $S_{\text{audit}}$ to the model owner
2. Model owner sends cryptographic commitments to its model $\mathsf{com}_h \leftarrow \mathsf{Commit}(h)$ and to the training data $\mathsf{com}_{\text{train}} \leftarrow \mathsf{Commit}(S_{\text{train}})$ to the auditor
3. They interact to execute ZKP: $b \leftarrow \langle \mathcal{P}(h, S_{\text{train}}), \mathcal{V} \rangle (\mathsf{com}_h, \mathsf{com}_{\text{train}}, S_{\text{audit}})$, where Model owner plays $\mathcal{P}$ and the auditor plays $\mathcal{V}$ and outputs $b$.

If the output is 1, the auditor is convinced that $f(h, S_{train}, S_{audit}) = 1$, where $h$ and $S_{train}$ are the model and training data committed in $\mathsf{com}_h$ and $\mathsf{com}_{train}$. Depending on $f$, some steps may be omitted; e.g., for an audit that checks accuracy or demographic parity on $S_{\text{audit}}$, $\mathsf{com}_{train}$ is unnecessary (see examples of $f$ in § A.1). Further, in some works, e.g., Shamsabadi et al. [2022], the model owner, rather than the auditor, samples the audit dataset.

## 4 Attacking ML Certification

Returning to our example, suppose the insurance company saves costs by *denying* claims. Intuitively, an accuracy audit with provable guarantees—such as those provided by zk proof-based systems—and with a sufficiently high threshold (e.g., passing only if accuracy on the auditor's dataset exceeds 95%) should prevent the company from deploying a model that unjustifiably denies too many claims.

[2]The commitment may be signed by the auditor.

We show that this intuition is false. Because machine learning is inherently data-dependent, certified properties need not hold once the model is deployed and applied to fresh data, even when drawn from the same distribution. More formally, while prior works certify that

$$f(h, S_{\text{train}}, S_{\text{audit}}) = 1$$

for some predicate $f$, a given model $h$, training data $S_{\text{train}}$, and audit dataset $S_{\text{audit}}$, this does not imply that the stronger property $F$ such that

$$F(h, S_{\text{train}}) = 1 \iff \Pr_{S_{\text{test}} \leftarrow \mathcal{D}}[f(h, S_{\text{train}}, S_{\text{test}}) = 1] > \mathsf{p}$$

where $\mathsf{p}$ is a non-negligible probability and $\mathcal{D}$ is a distribution over the entire population $Q = \{(x_i, y_i)\}_{i=1}^m$. The true goal of an audit, however, is precisely such stronger guarantees: an auditor typically seeks to ensure that a model remains fair, accurate, or robust not only on a particular dataset, but also on the unseen datasets it will encounter during deployment.

We show that this gap can be exploited. In particular, if $S_{\text{audit}}$ is known to the model provider before it is required to cryptographically commit to the model, the provider can ensure $f(h, S_{\text{train}}, S_{\text{audit}}) = 1$ (and thus pass the audit), *without* additionally satisfying $F$, which is the actual intended security property. A malicious model provider has strong incentives to do so: for example, the insurance company could deploy a model that maximizes accuracy on the audit dataset (and thus passes the audit), yet still unjustifiably denies numerous insurance claims.

**Attack Game with Known Audit Data** Before providing a concrete attack example, we introduce a theoretical tool – an attack game – which showcases the gap between verifying $f(h, S_{train}, S_{audit})$ (which is what prior approaches certified) and $F(h) = (\Pr_{S_{test} \leftarrow D}[f(h, S_{train}, S_{test}) = 1] > \mathsf{p})$ (the intuitive property that one would want to ensure) for audit schemes where the model owner is given the audit dataset at the beginning of the audit process.

For simplicity, we will assume that the audit process verifying $f(h, S_{train}, S_{audit})$ is perfectly secure, i.e., the outcome of $\langle \mathcal{P}(h, S_{\text{train}}), \mathcal{V} \rangle(\mathsf{com}_h, \mathsf{com}_{\text{train}}, S_{\text{audit}})$, where $\mathsf{com}_{train}$ is a commitment to $S_{train}$ and $\mathsf{com}_h$ is a commitment to $h$, is 1 if and only if $f(h, S_{train}, S_{audit}) = 1$.

In the game, the adversary will win only if it can come up with a model $h$ and training data $S_{train}$, such that: **(1)** $f(h, S_{train}, S_{audit}) = 1$, i.e, the adversary would pass an audit on the dataset $S_{audit}$, and **(2)** $F(h, S_{train}) = 0$. To make the attack even stronger, we require the adversary to additionally satisfy a utility requirement (formalized via a predicate $L$) in order to win the game. Intuitively, the goal of $L$ is to capture the actual intent of the malicious model owner: For example, in case of the insurance company that wishes to deny claims, we could use $L(h) = \Pr_{x \sim \{0,1\}^d}[h(x) = 0] > 0.9$.

**Definition 3** (Adaptive Training with Known Auditing Data)**.** *Let $f : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ be a predicate verified by the model certification, and let $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ be the actual intended security property. Let $\mathcal{X}$ be the feature space and $\mathcal{D}$ be a distribution over $\mathcal{X}$. Let $L$ denote the utility predicate*[3]*. Consider the following game played by an adversary $\mathcal{A}$:*

1. *Sample $S_{audit} \sim \mathcal{D}$*
2. *Given $S_{audit}$, $\mathcal{A}$ outputs a hypothesis $h_A$ and a training dataset $S_{train}$*
3. *Obtain $b = f(h_A, S_{train}, S_{audit})$*
4. *The output of the game is 1 ($\mathcal{A}$ 'wins') iff $b = 1$, $F(h_A, S_{train}) = 0$, and $L(h_A, S_{train}) = 1$. The output is 0 ($\mathcal{A}$ 'loses') otherwise.*

### 4.1 Example of a Data Forging Attack

We now give a concrete example of an attack within the framework of Def. 3 for the proofs of accuracy (e.g., [Zhang et al., 2020]) which utilize a dataset known to the model provider.

We consider our running example of an insurance company audit. Say the company uses a *decision tree* model (see §A.3 for background). The auditor wishes to check that the model is highly accurate,

[3]We assume that distribution $\mathcal{D}$ is implicitly "known" to $L$ (it is either hard-coded or provided as a parameter to $\mathcal{L}$). For simplicity of notation, we omit $\mathcal{D}$ from the description of $\mathcal{L}$.

i.e., $F(h) = \Pr_{x \sim \{0,1\}^d}[h(x) = y_x] > 0.95$, where $y_x$ is the true label of $x$. The insurance company saves on costs when it denies claims, i.e., the model provider's utility $L(h) = (\Pr_{x \sim \{0,1\}^d}[h(x) = 0] > 0.9)$.[4] Following Def. 3, given audit dataset $S_{audit}$, we wish to construct a hypothesis $h_A$ that passes the audit, while simultaneously achieving $F(h_A, S_{train}) = 0$ and $L(h_A, S_{train}) = 1$.

**Attack Outline** At a high level, we obtain $h_A$ from *honestly* training a model on a dataset $S'_{train}$ crafted with the knowledge of $S_{audit}$ in mind. In more detail, we build $S'_{train}$ from the audit dataset $S_{audit}$ using the corresponding true labels—to ensure the model passes the accuracy test on $S_{audit}$—and augment it with points in the neighborhood of those audit points. The points in the neighborhood are labeled 0, so that outside the audit samples the model learns to label every point with a 0. More formally, to construct $S'_{train}$, for every point $r \in S_{audit}$ and every (numeric) feature $i$, we add $r + \varepsilon\vec{b}_i$ with label 0, where $\varepsilon$ is some small number and $\vec{b}_i$ is the $i$th basis vector. We also add every $r \in S_{audit}$ to $S'_{train}$ with honest labels (see Algorithm 1 for formal description). Then, we train a tree on $S'_{train}$ via a standard DT training algorithm **until leaves are homogeneous**.

**Algorithm 1** Data Forging Attack

**Input**: Audit set $S_{audit}$, dimension $d$, $\varepsilon > 0$
**Output**: Training data $S'_{train}$

```
function ATTACK(S_audit, d, ε, g)
    S'_train ← S_audit
    for r ∈ S_audit do
        for i ∈ [d] do
            r_0 ← (r + ε b_i, 0)          ▷ b_i is the one-hot vector in dimension i
            r_1 ← (r − ε b_i, 0)
            S'_train ← S'_train ∪ {r_0, r_1}
    return S'_train
```

As we confirm in Fig. 5, this attack achieves good results. We now show that this attack allows the adversary to win in the attack game of Def 3. To this end, we first state the following theorem:

**Theorem 1.** *Classical decision tree training until homogeneity on the output of Algorithm 1 yields a tree $\mathcal{T}$ such that for every $x \in \mathbb{R}^d$, $\mathcal{T}(x) = 1$ only if $||x - r||_\infty < \varepsilon$ for some $r \in S_{audit}$.*

At a high level, the proof shows that if two points land in the same leaf, then any point lying between them on one coordinate must also fall in that leaf. Further, as the only non-zero points in $S'_{train}$ are audit points, every non-zero-labeled leaf contains an audit point. For any $x$ at least $\varepsilon$ away from all audit points, if $\mathcal{T}(x) = 1$, one can construct a nearby training point with label 0 that must lie in the same leaf, giving a contradiction as we trained until homogeneity. See §B.1 for details. □

Thus, whenever a model provider generates a training dataset using Algorithm 1, an honestly trained decision tree that grows until homogeneity will achieve perfect accuracy on the audit dataset, yet predict zero for all inputs that lie outside an $\varepsilon$-neighborhood of the audit dataset points. Thus, for an appropriate choice of epsilon, the adversary wins in the game specified in Definition 3 with probability one.[5]

**Detection** Attacks by malicious model providers can typically be prevented by requiring cryptographic proofs that a model has been trained using a specific (benign) algorithm. However, such proofs are useless here: The model provider trains the model *honestly* using a standard training algorithm, and it's the training data itself that enables the attack. One might expect training data manipulation to be caught by statistical tests, e.g., Welch's $t$-test, which assess whether two samples are drawn from the same distribution. As we show in §C, however, this is not the case. We can cause the distributional properties of the training data to converge towards the audit data without sacrificing the efficacy of this attack by adding more copies of the audit data to the training data. This causes the audit data and the training data to appear as if they were drawn from the same distribution under a variety of statistical tests without impacting the model's ability to learn the desired behavior.

[4] For simplicity, we consider datapoints in $\{0, 1\}^d$

[5] Assuming that a model which almost always outputs 0 is not highly accurate in our scenario.

## 5 Evaluation

In this section we underscore the importance of data forging attacks by mounting proof of concept attacks for models trained on a variety of datasets. We show that our attack is effective in making inaccurate models appear accurate and unfair models appear fair, and empirically demonstrate a variety of other qualities, e.g. undetectability with a variety of statistical tools.

**Experimental Setup.** We use six well-known fairness benchmarking datasets in our experiments: ACSEmployment Ding et al. [2021], Adult Becker and Kohavi [1996], COMPAS Angwin et al. [2016], German Credit Hofmann [1994], Default Credit Yeh and hui Lien [2009], and Communities & Crime Redmond [2009]. We implemented a modified version of our attack from §4.1 in Python 3.12.3 using SciKit-Learn version 1.6.1 that attempts to minimize an objective when deployed and evaluated its performance against a variety of datasets. For a given run, we split the dataset into an evaluation dataset consisting of $30\%$ of the data, an audit dataset containing 1000 data points, and an initial training data set. We represent the interpolation between a fully honest training run and a fully malicious one by the *attack parameter*, which takes a value between 0 (fully honest) and 1 (fully malicious). The attack parameter controls what proportion of audit data points are included in the training data and what proportion of the initial training data is labeled maliciously. The specifics of how honest and malicious data points are labeled depends on the objective.

To attack accuracy, we constructed a training dataset using a modified Algorithm 1, labeling the additional data from the attack with $1 - r_y$ rather than $0$ and adding them to the initial training data set. For our fairness attack, we constructed the training data similarly, changing how honest and malicious data points were labeled. Honest data points were given random labels, while malicious data points were labeled according to their sensitive attribute. Both attacks then fit a decision tree to their constructed training data using SciKit-Learn's decision tree classifier class.

To ensure that our attack would evade statistical detection, we added extra copies of the audit data to the training data, as computed in Corollary 1 to pass Welch's $t$-test with significance level $0.05$.

**Attacking Accuracy Audits.** We ran our attack on six benchmark datasets – three shown in Figure 2 (remaining in Figure 4). Across all datasets, our data forging attack enforces high audit accuracy while simultaneously encouraging low performance on real-world evaluation data. Thus **our attack successfully makes inaccurate models *appear* accurate to an auditor.**

**Attacking Fairness Audits.** We also performed the attack while targeting demographic parity (using sex as the sensitive attribute) on three datasets, which we present in Figure 3. We were able to reliably train a model with close to $0$ fairness gap on the audit dataset, but close to $1$ fairness gap when deployed. In other words, **our attack successfully makes unfair models *appear* fair to an auditor.**

**Evading Detection via Statistical Methods.** We show how our attack can be executed in ways that evade detection by a variety of statistical approaches in Appendix Table 2. We were able to construct malicious training datasets with summary statistics that match those of the audit dataset very closely, and Welch's $t$-test and Levene's test regularly concluded that the audit and test datasets were drawn from the same distribution. This is consistent with our theoretical results in Appendix C.

**Additional Results.** An adversary can use data forging attacks to achieve concrete goals beyond degradation of accuracy or fairness, as we show in the Appendix G. For example, Figure 5 shows how an insurance provider could use our attack to hide the claim denial rate of a model from auditors. Figure 6 also shows preliminary results which suggest that our attack generalizes to neural networks.

## 6 Discussion and Future Work

This work brings attention to data-dependent vulnerabilities in cryptographic auditing methods for machine learning models. We propose an attack strategy that passes cryptographic certification while undermining the goals of those certifications for real-world performance. We then introduce new formal security definitions which address these vulnerabilities.

The attack strategy presented in this work poses several open questions. While we demonstrate the data forging attack is undetected even in the presence of Welch's $t$-test and Levene's test, it remains

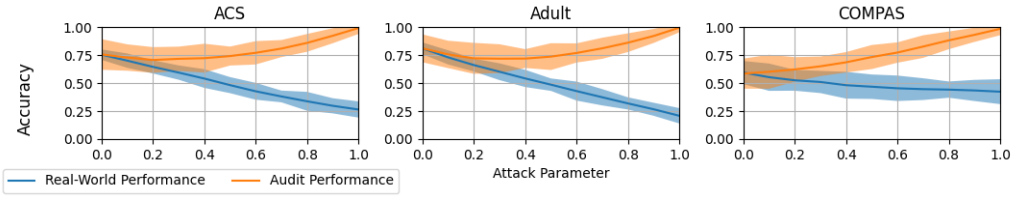


Figure 2: Performance of models trained on datasets constructed to minimize real-world accuracy while still passing an audit for several benchmarks. Values are averages over ten runs, error bars represent one standard deviation.

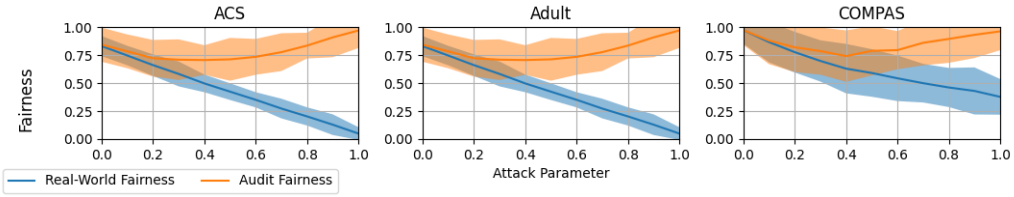


Figure 3: Fairness of models trained on constructed datasets using various benchmarks to target demographic parity. Values are averages over ten runs, error bars represent one standard deviation. Fairness means $1 -$ fairness gap.

to be seen whether other statistical tests could effectively detect the attack. Based on the results that we have derived, we find it unlikely that other statistical tests will be effective in detecting the attack. However, we reserve such analysis for future work. We provide rigorous formal proofs that our attacks are effective on decision trees, and preliminary evidence that a similar approach generalizes to neural networks. Characterizing a formal relationship between neural network model capacity and attack effectiveness could be a promising direction in future work.

Our findings underscore the importance of keeping audit data hidden until the service provider's model is committed. This imposes a limitation on auditing in practice: auditors must either regularly gather fresh data (since the audit dataset is typically revealed during the audit), use additional cryptographic techniques such as secure multiparty computation to keep data hidden during the audit, or perform continuous auditing on user data. Each of these options has strengths and drawbacks which should be evaluated in more detail by future work.

## Acknowledgments

We thank Nicolas Papernot for the contributions to this work, Chen-Da Liu-Zhang for many helpful discussions, and Jonas Guan, Sierra Wyllie and Mohammad Yaghini for helpful feedback on an earlier draft.

This paper was prepared in part for information purposes by the Artificial Intelligence Research group of JPMorgan Chase & Co and its affiliates ("JP Morgan"), and is not a product of the Research Department of JP Morgan. JP Morgan makes no representation and warranty whatsoever and disclaims all liability, for the completeness, accuracy or reliability of the information contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction, and shall not constitute a solicitation under any jurisdiction or to any person, if such solicitation under such jurisdiction or to such person would be unlawful.

Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute. The work was further supported by the CBI postdoctoral fellowship.

## References


K. Abbaszadeh, C. Pappas, J. Katz, and D. Papadopoulos. Zero-knowledge proofs of training for deep neural networks. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 4316–4330, 2024.

J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine bias. *ProPublica*, May 2016. Retrieved January 15, 2022.

M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar. The security of machine learning. *Mach. Learn.*, 81(2):121–148, 2010.

B. Becker and R. Kohavi. Adult. UCI Machine Learning Repository, 1996.

J. G. Bourrée, H. Lautraite, S. Gambs, G. Trédan, E. L. Merrer, and B. Rottembourg. P2NIA: privacy-preserving non-iterative auditing. *CoRR*, abs/2504.00874, 2025. doi: 10.48550/ARXIV.2504.00874. URL `https://doi.org/10.48550/arXiv.2504.00874`.

I. Chang, K. Sotiraki, W. Chen, M. Kantarcioglu, and R. Popa. {HOLMES}: Efficient distribution testing for secure collaborative learning. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 4823–4840, 2023.

F. Ding, M. Hardt, J. Miller, and L. Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing Systems*, 34, 2021.

V. Duddu, A. Das, N. Khayata, H. Yalame, T. Schneider, and N. Asokan. Attesting distributional properties of training data for machine learning. In *European Symposium on Research in Computer Security*, pages 3–23. Springer, 2024.

O. Franzese, A. S. Shamsabadi, and H. Haddadi. Oath: Efficient and flexible zero-knowledge proofs of end-to-end ml fairness. *arXiv preprint arXiv:2410.02777*, 2024.

S. Garg, A. Goel, S. Jha, S. Mahloujifar, M. Mahmoody, G.-V. Policharla, and M. Wang. Experimenting with zero-knowledge proofs of training. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 1880–1894, 2023.

S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In R. Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 291–304. ACM, 1985. doi: 10.1145/22145.22178. URL `https://doi.org/10.1145/22145.22178`.

M. Hardt. The emerging science of machine learning benchmarks. *Manuscript. https://mlbenchmarks.org*, 2025.

H. Hofmann. Statlog (German Credit Data). UCI Machine Learning Repository, 1994. DOI: https://doi.org/10.24432/C5NC77.

D. Kang, T. Hashimoto, I. Stoica, and Y. Sun. Scaling up trustless DNN inference with zero-knowledge proofs. *CoRR*, abs/2210.08674, 2022. doi: 10.48550/ARXIV.2210.08674. URL `https://doi.org/10.48550/arXiv.2210.08674`.

T. Liu, X. Xie, and Y. Zhang. zkcnn: Zero knowledge proofs for convolutional neural network predictions and accuracy. In Y. Kim, J. Kim, G. Vigna, and E. Shi, editors, *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 2968–2985. ACM, 2021. doi: 10.1145/3460120.3485379. URL `https://doi.org/10.1145/3460120.3485379`.

M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 220–229, 2019.

C. Pappas and D. Papadopoulos. Sparrow: Space-efficient zksnark for data-parallel circuits and applications to zero-knowledge decision trees. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 3110–3124, 2024.

M. Redmond. Communities and Crime. UCI Machine Learning Repository, 2009.

P. Saleiro, B. Kuester, L. Hinkson, J. London, A. Stevens, A. Anisfeld, K. T. Rodolfa, and R. Ghani. Aequitas: A bias and fairness audit toolkit. *arXiv preprint arXiv:1811.05577*, 2018.

A. S. Shamsabadi, S. C. Wyllie, N. Franzese, N. Dullerud, S. Gambs, N. Papernot, X. Wang, and A. Weller. Confidential-profitt: confidential proof of fair training of trees. In *The Eleventh International Conference on Learning Representations*, 2022.

A. S. Shamsabadi, G. Tan, T. I. Cebere, A. Bellet, H. Haddadi, N. Papernot, X. Wang, and A. Weller. Confidential-dpproof: Confidential proof of differentially private training. In *International Conference on Learning Representations (ICLR)*, 2024.

J. Steinhardt, P. W. Koh, and P. Liang. Certified defenses for data poisoning attacks. In *NIPS*, pages 3517–3529, 2017.

H. Sun, T. Bai, J. Li, and H. Zhang. Zkdl: Efficient zero-knowledge proofs of deep learning training. *IEEE Transactions on Information Forensics and Security*, 2024.

S. Tan, R. Caruana, G. Hooker, and Y. Lou. Distill-and-compare: Auditing black-box models using transparent model distillation. In *AIES*, pages 303–310. ACM, 2018.

F. Tramer, V. Atlidakis, R. Geambasu, D. Hsu, J.-P. Hubaux, M. Humbert, A. Juels, and H. Lin. Fairtest: Discovering unwarranted associations in data-driven applications. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 401–416. IEEE, 2017.

H. Wang and T. Hoang. ezdps: An efficient and zero-knowledge machine learning inference pipeline. *Proc. Priv. Enhancing Technol.*, 2023(2):430–448, 2023. doi: 10.56553/POPETS-2023-0061. URL `https://doi.org/10.56553/popets-2023-0061`.

B. L. Welch. The generalization of 'student's'problem when several different population varlances are involved. *Biometrika*, 34(1-2):28–35, 1947.

C. Yadav, A. R. Chowdhury, D. Boneh, and K. Chaudhuri. Fairproof: Confidential and certifiable fairness for neural networks. *arXiv preprint arXiv:2402.12572*, 2024.

I.-C. Yeh and C. hui Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2, Part 1): 2473–2480, 2009. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2007.12.020. URL `https://www.sciencedirect.com/science/article/pii/S0957417407006719`.

G. Zhang, F. E. Dorner, and M. Hardt. How benchmark prediction from fewer data misses the mark. *arXiv preprint arXiv:2506.07673*, 2025a.

J. Zhang, Z. Fang, Y. Zhang, and D. Song. Zero knowledge proofs for decision tree predictions and accuracy. In *CCS*, pages 2039–2053. ACM, 2020.

T. Zhang, S. Dong, O. D. Kose, Y. Shen, and Y. Zhang. Fairzk: A scalable system to prove machine learning fairness in zero-knowledge. In M. Blanton, W. Enck, and C. Nita-Rotaru, editors, *IEEE Symposium on Security and Privacy, SP 2025, San Francisco, CA, USA, May 12-15, 2025*, pages 3460–3478. IEEE, 2025b. doi: 10.1109/SP61157.2025.00205. URL `https://doi.org/10.1109/SP61157.2025.00205`.

# A Additional Preliminaries

## A.1 Example of Auditing Predicates

**Auditing Accuracy** To audit accuracy, we consider the empirical accuracy as follows:

$$\hat{\ell}_S(h) = \frac{1}{n} \sum_{(x,y)\in S} \mathbb{I}(h(x) \neq y)$$

**Algorithm 2** Welch's $t$-test

**Input:** $\mathcal{X} = \{x_i\}_{i\in[n]}$, $\mathcal{Y} = \{y_i\}_{i\in[m]}$, where $x_i \sim X$ and $y_i \sim Y$, and a significance level $\alpha$
**Output**: Null hypothesis $H_0$ (i.e., $\mu_X = \mu_Y$) or alternative hypothesis $H_1$ (i.e., $\mu_X \neq \mu_Y$)

1: Compute sampled means $\bar{x} = \frac{\sum_i x_i}{n}$ and $\bar{y} = \frac{\sum_i y_i}{m}$
2: Compute sampled variances $v_x = \frac{\sum_i (\bar{x}-x_i)^2}{n-1}$ and $v_y = \frac{\sum_i (\bar{y}-y_i)^2}{m-1}$.
3: Compute the test statistic $t = \frac{\bar{x}-\bar{y}}{\sqrt{v_x/n+v_y/m}}$
4: Compute the degree of freedom $d = \frac{(g_x+g_y)^2}{g_x^2/(n-1)+g_y^2/(m-1)}$, where $g_x = v_x/n$ and $g_y = v_y/m$
5: Obtain the critical value $t_{cr}$ from the $t$-table, given $d$ and $\alpha$.
6: **If** $|t| < t_{cr}$ **return** $H_0$ **else return** $H_1$

where $n = |S|$, and define the empirical predicate $f$ as follows:

$$f(h, S_{\text{audit}}) = 1 \iff \hat{\ell}_S(h) \leq t$$

**Auditing Fairness with Demographic Parity** Demographic parity is one of the most basic fairness metrics, measuring the difference between the prediction probabilities conditioned on a sensitive attribute. We consider the empirical parity differences as follows:

$$\Delta_{\mathsf{dp}}(h, S_{\text{audit}}) = \left| \frac{1}{n_0} \sum_{x\in S_0} \mathbb{I}(h(x) = 1) - \frac{1}{n_1} \sum_{x\in S_1} \mathbb{I}(h(x) = 1) \right|$$

where $s_x$ denotes the sensitive feature of a data point $x$, $S_0 = \{x \in S_{\text{audit}} : s_x = 0\}$, $S_1 = \{x \in S_{\text{audit}} : s_x = 1\}$, $n_0 = |S_0|$, and $n_1 = |S_1|$. To audit fairness w.r.t a model $h$ and a dataset $S_{\text{audit}}$, we define the corresponding empirical predicate $f$ as follows.

$$f(h, S_{\text{audit}}) = 1 \iff \Delta_{\mathsf{dp}}(h, S_{\text{audit}}) \leq t$$

### A.2 Welch's $t$-test

**Welch's $t$-test** The goal of $t$-test is to determine whether the unknown population means of two groups are equal or not. That is, for random variables $X$ and $Y$, it compares the following hypotheses on their means $\mu_X = \mathrm{E}[X]$ and $\mu_Y = \mathrm{E}[Y]$:

- Null Hypothesis $H_0$: $\mu_X = \mu_Y$
- Alternative Hypothesis $H_1$: $\mu_X \neq \mu_Y$

Assuming that $X$ and $Y$ independently follow Gaussian distributions with unknown variances, Welch's $t$-test proceeds as in Algorithm 2.

### A.3 Decision Trees

In our attack constructions we focus on decision tree models. Decision tree-based solutions are among the most popular machine learning algorithms, particularly known for their effectiveness in classification problems such as loan approval and fraud detection. A decision tree is trained by recursively partitioning the dataset from the root to the leaves. At each step, a split is determined by a splitting rule that aims to maximize an objective function, such as information gain. For prediction, the input follows a path from the root to a leaf, where at each internal node, the decision depends on whether the input satisfies the corresponding threshold (see Algorithm 3).

For completeness, in Algorithm 3 we present the algorithm for decision tree inference.

### A.4 Security Properties of Zero-Knowledge Proofs

Let $\mathsf{ZKP} = (\mathcal{P}, \mathcal{V})$ be an interactive proof system for a relation $\mathcal{R} = \bigcup_{\lambda\in\mathbb{N}} \mathcal{R}_\lambda$. In what follows, we denote by PPT *probabilistic polynomial time*.

**Algorithm 3** Decision Tree Inference

**Input:** Decision tree $h$, input $\mathbf{a}$.
**Output**: Classification result.

```
Let cur := h.root                                  ▷ Set cur to be root of the tree
while cur is not a leaf do
    if a[cur.attr] < cur.thr then
        cur := cur.left.                           ▷ Set cur to be current node's left child
    else
        cur := cur.right.                          ▷ Set cur to be current node's right child
return cur.class
```

**Completeness** ZKP is (perfectly) *complete* if for any $(\mathsf{x}, \mathsf{w})$ satisfying $\mathcal{R}$, it holds that:

$$\Pr[1 \leftarrow \langle \mathcal{P}(\mathsf{w}), \mathcal{V} \rangle(\mathsf{x})] = 1.$$

**Knowledge Soundness** ZKP is (adaptively) *knowledge sound* with knowledge error $\kappa$ if for any (stateful) PPT adversary $\mathcal{P}^* = (\mathcal{P}_0, \mathcal{P}_1)$, there exists an expected polynomial time extractor $\mathcal{E}$ such that the following holds:

$$p_{\text{ext}} \geq p_{\text{acc}} - \kappa$$

where

$$p_{\text{ext}} = \Pr\left[R_\lambda(\mathsf{x}, \mathsf{w}) = 1 \,:\, \mathsf{x} \leftarrow \mathcal{P}_0(1^\lambda); \mathsf{w} \leftarrow \mathcal{E}_{\mathcal{P}}(\mathsf{x})\right]$$
$$p_{\text{acc}} = \Pr\left[b = 1 \,:\, \mathsf{x} \leftarrow \mathcal{P}_0(1^\lambda); b \leftarrow \langle \mathcal{P}_1, \mathcal{V} \rangle(\mathsf{x})\right]$$

where $\mathcal{E}$ has non-black-box access to $\mathcal{P}^*$. Informally, this means that any cheating prover must know a valid witness if it convinces verifier.

**Zero-Knowledge** Let $\mathsf{view}_{\mathcal{V}}^{\mathcal{P}(\mathsf{w})}(\mathsf{x})$ be a string consisting of all the incoming messages that $\mathcal{V}$ receives from $\mathcal{P}$ during the interaction $\langle \mathcal{P}(\mathsf{w}), \mathcal{V} \rangle(\mathsf{x})$, and $\mathcal{V}$'s random coins. $\Pi$ is (honest verifier) *zero-knowledge* if there exists a PPT simulator $\mathcal{S}$ such that for any adversary $\mathcal{A}$ and any $(\mathsf{x}, \mathsf{w}) \in \mathcal{R}_\lambda$, the following is negligible in $\lambda$.

$$\left| \Pr\left[b = 1 \,:\, b \leftarrow \mathcal{A}(\mathsf{view}_{\mathcal{V}}^{\mathcal{P}(\mathsf{w})}(\mathsf{x})) \quad \right] - \quad \Pr\left[b = 1 \,:\, \mathsf{view}' \leftarrow \mathcal{S}(\mathsf{x}); \quad b \leftarrow \mathcal{A}(\mathsf{view}')\right] \right|$$

Informally, this means that the protocol execution reveals no information to $\mathcal{V}$ about $\mathsf{w}$.

### A.5 Security Properties of Commitment Schemes

Let Commit be a commitment scheme. For simplicity, we omit the key generation algorithm Gen for simplicity and present a class of the simplest commitments whose openings are checked by re-computing and comparing (e.g., hash commitment $H(m||\rho)$). More generally, some commitment schemes require a separate verification algorithm Verify to check the validity of a commitment given some *decommitment* information. Our auditing framework can be extended to such schemes by having the model provider prove the knowledge of the decommitment information in zero knowledge.

**Binding** Commit is *computationally binding* if for any PPT adversary $\mathcal{A}$, the following is negligible in $\lambda$:

$$\Pr\left[\mathsf{Commit}(m; \rho) = \mathsf{Commit}(m'; \rho') \wedge m \neq m' \,:\, (m, m', \rho, \rho') \leftarrow \mathcal{A}(1^\lambda)\right]$$

**Hiding** Commit is *computationally hiding* if for any PPT adversary $\mathcal{A}$, the following is negligible in $\lambda$:

$$\left| \Pr\left[b = 1 \,:\, \begin{array}{r} m_0, m_1 \leftarrow \mathcal{A}(1^\lambda); \rho \leftarrow \{0,1\}^{\ell_r(\lambda)}; \\ \mathsf{com} \leftarrow \mathsf{Commit}(m_b; \rho); b \leftarrow \{0,1\}; b' \leftarrow \mathcal{A}(\mathsf{com}) \end{array}\right] - \frac{1}{2} \right|$$

## B Deferred Proofs

### B.1 Proof of Theorem 1

*Proof.* First, let us show that if points $a$ and $b$, where $a_i < c_i < b_i$ and $c_j = b_j$ for all $j \neq i$, get sorted into the same decision tree leaf, then $c$ is also sorted into that same leaf. Notice that in order for two points $x$ and $y$ to get sorted into different leaves, there must be a node which splits on a feature $i$ such that $x_i \neq y_i$ and $x_i \leq t \leq y_i$ or $y_i \leq t \leq x_i$ where $t$ is the threshold to split upon. Then if $b$ and $c$ were sorted into separate leaves, there must be a node on the path that $b$ takes through the tree that splits on feature $i$ with a threshold $t$ that satisfies $c_i \leq t \leq b_i$. However, such a node would also sort $a$ distinctly from $b$, so such a node cannot occur. Thus, $b$ and $c$ must be sorted into the same leaf.

Now, note that as we train the tree until its leafs are homogeneous, every datapoint in $S'_{train}$ must be classified correctly (according to the label we assigned to it in Algorithm 1). Further, since the only datapoints in $S'_{train}$ with non-zero labels are datapoints from $S_{audit}$, for every leaf in $\mathcal{T}$ that is associated with a non-zero class, we have at least one $r \in S_{audit}$ that gets sorted into this leaf.

Consider $x \in \mathbb{R}^d$ such that $||x - r||_\infty \geq \varepsilon$ for all $r \in S_{audit}$. Say $\mathcal{T}(x) = 1$, i.e., there exists a leaf such that $x$ belongs to this leaf and the leaf corresponds to class one. Consider $r \in S_{audit}$ that belongs to this leaf (by above, such $r$ exists). By the definition of the L-infinity norm there exists some dimension $i$ where $|r_i - x_i| > \varepsilon$. Suppose $r_i - x_i > \varepsilon$. Notice that there is a point $r - \varepsilon\vec{b}_i \in S'_{train}$ which satisfies that $(r - \varepsilon\vec{b}_i)_j = r_j$ for all $j \neq i$, and where $x_i < (r - \varepsilon\vec{b}_i)_i < r_i$. Then by above, $r - \varepsilon\vec{b}_i$ must be sorted into the same leaf as $r$ and $x$. But $r - \varepsilon\vec{b}_i$ has label $g(r) = 0$, while for $x$ holds $\mathcal{T}(x) = 1$. Thus, we found a contradition. The same argument holds if $x_i - r_i > \varepsilon$, but using the point $r + \varepsilon\vec{b}_i$ instead of $r - \varepsilon\vec{b}_i$. □

## C Attack Detection

While proof of training alone cannot detect the attack above (as it relies on training the decision tree entirely honestly), nor can a black-box audit where the model owner knows the audit data before training time, we might still hope to detect when these attacks occur. For example, we might hope to conduct statistical tests on the training data to determine if it was honestly sampled from the underlying distribution or if it was adversarially constructed. In such a case, we cannot directly compare the training data to the true distribution of real data because the underlying distribution is not fully known to the auditor. Instead, we must compare the training data with a sample from that distribution. In the most simple case, this sample is the reference set $S_{audit}$.

We argue that under a certain family of functions, our constructed training set is indistinguishable from $S_{audit}$.

**Definition 4.** *Suppose $\vec{\alpha}$ is a set of bins over $d$ dimensions. Then $H_{\vec{\alpha}} : (\mathbb{R}^d \times \{0,1\})^* \to \mathcal{H}$ is the function which takes databases over $d$ features and a binary classification to their normalized histogram with bins $\vec{\alpha}$.*

**Definition 5.** *A function $f : (\mathbb{R}^d \times \{0,1\})^* \to \mathbb{R}$ is called $(\gamma, c)$-magnitude insensitive if there exists a choice of bins $\vec{\alpha}$ and function $f' : \mathcal{H} \to \mathbb{R}$ such that $|f(D) - f'(H_{\vec{\alpha}}(D))| < \gamma$ for all $D \in (\mathbb{R}^d \times \{0,1\})^*$ and $|f'(H_{\vec{\alpha}}(D)) - f'(H_{\vec{\alpha}}(D||r))| \leq \frac{c}{|D|}$ for all $D \in (\mathbb{R}^d \times \{0,1\})^*$ and $r \in \mathbb{R}^d \times \{0,1\}$.*

**Theorem 2.** *If $f$ is $(\gamma, c)$-magnitude insensitive, then $|f(S_{audit}) - f\left(S^k_{audit}||\delta\right)| \leq \varepsilon$ for any $\varepsilon > 2\gamma$ and $k \geq \frac{2dc}{\varepsilon - 2\gamma}$, where $\delta$ is the additional training data created by Algorithm 1 when run with input $S_{audit}, d, \varepsilon, g$ for any $g$.*

*Proof.* We will write $f'$ to be the $\gamma$-approximation of $f$ guaranteed to exist by the fact that $f$ is $(\gamma, c)$-magnitude insensitive. Observe that because $H_{\vec{\alpha}}$ takes databases to their normalized histograms, $H_{\vec{\alpha}}(S_{audit}) = H_{\vec{\alpha}}\left(S^k_{audit}\right)$, because the non-normalized histograms of the two databases are simply scaled versions of one another.

Next, it will be helpful to show that for any two databases $D_1, D_2 \in (\mathbb{R}^d \times \{0,1\})^*$, we have $|f'(H_{\vec{\alpha}}(D_1)) - f'(H_{\vec{\alpha}}(D_1||D_2))| \leq c\frac{|D_2|}{|D_1|}$. Let us write $D_2 = d_1||d_2||\ldots||d_{|D_2|}$. Then we get that

$$\begin{aligned}
&|f'(H_{\vec{\alpha}}(D_1)) - f'(H_{\vec{\alpha}}(D_1||D_2))| \\
&= |f'(H_{\vec{\alpha}}(D_1)) - f'(H_{\vec{\alpha}}(D_1||d_1)) + f'(H_{\vec{\alpha}}(D_1||d_1)) - \ldots \\
&\qquad + f'(H_{\vec{\alpha}}(D_1||d_1||d_2||\ldots||d_{|D_2|-1})) - f'(H_{\vec{\alpha}}(D_1||D_2))| \\
&\leq |f'(H_{\vec{\alpha}}(D_1)) - f'(H_{\vec{\alpha}}(D_1||d_1))| + |f'(H_{\vec{\alpha}}(D_1||d_1)) - f'(H_{\vec{\alpha}}(D_1||d_1||d_2)| + \ldots \\
&\qquad + |f'(H_{\vec{\alpha}}(D_1||d_1||d_2||\ldots||d_{|D_2|-1})) - f'(H_{\vec{\alpha}}(D_1||D_2))| \\
&\leq \frac{c}{|D_1|} + \frac{c}{|D_1|+1} + \ldots + \frac{c}{|D_1|+|D_2|-1} \\
&\leq c\frac{|D_2|}{|D_1|}
\end{aligned}$$

Then we can apply this to $S^k_{audit}$ and $S^k_{audit}||\delta$; recall that $|\delta| = 2d|S_{audit}|$. Then we see that

$$\begin{aligned}
\left|f'\left(H_{\vec{\alpha}}\left(S_{audit}\right)\right) - f'\left(H_{\vec{\alpha}}\left(S^k_{audit}||\delta\right)\right)\right| &= \left|f'\left(H_{\vec{\alpha}}\left(S^k_{audit}\right)\right) - f'\left(H_{\vec{\alpha}}\left(S^k_{audit}||\delta\right)\right)\right| \\
&\leq c\frac{2d|S_{audit}|}{k|S_{audit}|} \\
&\leq c\frac{2d}{\left(\frac{2dc}{\varepsilon-2\gamma}\right)} = \varepsilon - 2\gamma
\end{aligned}$$

We have two cases now.

Case 1: $f'\left(H_{\vec{\alpha}}\left(S_{audit}\right)\right) \geq f'\left(H_{\vec{\alpha}}\left(S^k_{audit}||\delta\right)\right)$. Then we have

$$\begin{aligned}
\varepsilon - 2\gamma &\geq f'\left(H_{\vec{\alpha}}\left(S_{audit}\right)\right) - f'\left(H_{\vec{\alpha}}\left(S^k_{audit}||\delta\right)\right) \\
&= f(S_{audit}) - f(S_{audit}) + f'(H_{\vec{\alpha}}(S_{audit})) \\
&\qquad - f(S^k_{audit}||\delta) + f(S^k_{audit}||\delta) - f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) \\
&\geq f(S_{audit}) - |f(S_{audit}) - f'(H_{\vec{\alpha}}(S_{audit}))| \\
&\qquad - f(S^k_{audit}||\delta) - |f(S^k_{audit}||\delta) - f'(H_{\vec{\alpha}}(S^k_{audit}||\delta))| \\
&\geq f(S_{audit}) - \gamma - f(S^k_{audit}||\delta) - \gamma
\end{aligned}$$

and so we see that $\varepsilon \geq f(S_{audit}) - f(S^k_{audit}||\delta)$. We also have

$$\begin{aligned}
f(S_{audit}) - f(S^k_{audit}||\delta) &= f'(H_{\vec{\alpha}}(S_{audit})) - f'(H_{\vec{\alpha}}(S_{audit})) + f(S_{audit}) \\
&\qquad - f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) + f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) - f(S^k_{audit}||\delta) \\
&\geq f'(H_{\vec{\alpha}}(S_{audit})) - |f'(H_{\vec{\alpha}}(S_{audit})) - f(S_{audit})| \\
&\qquad - f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) - |f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) - f(S^k_{audit}||\delta)| \\
&\geq f'(H_{\vec{\alpha}}(S_{audit})) - \gamma - f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) - \gamma \\
&\geq -2\gamma \\
&> -\varepsilon
\end{aligned}$$

Then $|f(S_{audit}) - f(S^k_{audit}||\delta)| \leq \varepsilon$.

Case 2: $f'(H_{\vec{\alpha}}(S_{audit})) \leq f'(H_{\vec{\alpha}}(S^k_{audit}||\delta))$. Then we have

$$\begin{aligned}
\varepsilon - 2\gamma &\geq f'\left(H_{\vec{\alpha}}\left(S^k_{audit}||\delta\right)\right) - f'\left(H_{\vec{\alpha}}\left(S_{audit}\right)\right) \\
&= f(S^k_{audit}||\delta) - f(S^k_{audit}||\delta) + f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) \\
&\qquad - f(S_{audit}) + f(S_{audit}) - f'(H_{\vec{\alpha}}(S_{audit})) \\
&\geq f(S^k_{audit}||\delta) - |f(S^k_{audit}||\delta) - f'(H_{\vec{\alpha}}(S^k_{audit}||\delta))| \\
&\qquad - f(S_{audit}) - |f(S_{audit}) - f'(H_{\vec{\alpha}}(S_{audit}))| \\
&\geq f(S^k_{audit}||\delta) - \gamma - f(S_{audit}) - \gamma
\end{aligned}$$

and so we see that $\varepsilon \geq f(S^k_{audit}||\delta) - f(S_{audit})$. We also have

$$\begin{aligned}
f(S^k_{audit}||\delta) - f(S_{audit}) &= f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) - f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) + f(S^k_{audit}||\delta) \\
&\qquad - f'(H_{\vec{\alpha}}(S_{audit})) + f'(H_{\vec{\alpha}}(S_{audit})) - f(S_{audit}) \\
&\geq f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) - |f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) + f(S^k_{audit}||\delta)| \\
&\qquad - f'(H_{\vec{\alpha}}(S_{audit})) - |f'(H_{\vec{\alpha}}(S_{audit})) - f(S_{audit})| \\
&\geq f'(H_{\vec{\alpha}}(S^k_{audit}||\delta)) - \gamma - f'(H_{\vec{\alpha}}(S_{audit})) - \gamma \\
&\geq -2\gamma \\
&\geq -\varepsilon
\end{aligned}$$

Then $|f(S_{audit}) - f(S^k_{audit}||\delta)| \leq \varepsilon$. □

This theorem does not suggest that it is completely impossible to detect the attack given in Algorithm 1. Rather, it only precludes detection by a certain class of functions. However, we argue that this class is expansive and covers many intuitive approaches.

The sole requirement for the audit metric $f$ is that it must be approximable by $f'$ which satisfies three properties. Firstly, $f'$ operates over histograms for some choice of bins $\vec{\alpha}$. This is a necessary condition, as if $f$ were not approximable by a function over a binning of the training data, we could drastically change the audit outcome by simply adding a small amount of noise to the data. Next, $f'$ must be relatively insensitive to additional data. The intuition here is that no individual datapoint should dramatically change the outcome of the audit. Finally, $f'$ operates over normalized histograms. This property is necessary for the proof to go through, but is satisfied by many intuitive audit metrics. For example, the mean and standard deviation of a feature (even conditioned on any arbitrary set of features) are approximable from a normalized histogram.

**Lemma 1.** *Let $\mu_j(D)$ be the mean of (bounded) feature $j$ of a dataset $D$. Then for every $\gamma > 0$, $\mu_j(D)$ is $(\gamma, M - m)$-magnitude insensitive, where $B$ is the set of bins in the histogram and $M, m$ are an upper and lower bound on possible $j$-values respectively.*

*Proof.* Notice that $\mu_j(D) \approx \sum_{i \in B} p_i x_{j,i}$ where $B$ is the set of bins in the histogram, $p_i$ is the height of bin $i$ in the normalized histogram of $D$, and $x_{j,i}$ is the $j$-value of bin $i$. Let us show that for any $\gamma > 0$, there exists a binning of the data such that this is a $\gamma$-approximation of $\mu_j(D)$. Let the bins in feature $j$ have width $\gamma$. Then for each datapoint $d$ with $j$ value $j_d$, bin $i$, and binned $j$-value $x_{j,i}$, we have that $|x_{j,i} - j_d| \leq \gamma$. Then

$$\begin{aligned}
\sum_{i \in B} p_i x_{j,i} &= \sum_{i \in B} \frac{c_i}{|D|} x_{j,i} \\
&= \sum_{d \in D} \frac{1}{|D|} x_{j,i} \\
\implies \left| \sum_{i \in B} p_i x_{j,i} - \sum_{d \in D} \frac{1}{|D|} j_d \right| &= \left| \sum_{d \in D} \frac{1}{|D|} x_{j,i} - \sum_{d \in D} \frac{1}{|D|} j_d \right| \\
&= \left| \frac{1}{|D|} \sum_{d \in D} (x_{j,i} - j_d) \right| \\
&\leq \frac{1}{|D|} \sum_{d \in D} |x_{j,i} - j_d| \\
&\leq \frac{1}{|D|} \sum_{d \in D} \gamma \\
&= \gamma
\end{aligned}$$

Next, let us show that the sensitivity of our approximation of $\mu_j$ is upper bounded by $\frac{M-m}{|D|}$. Notice that by adding a single point, one histogram bin will increase by 1 and the rest will be unchanged.

Then for every bin $k$,

$$\begin{aligned}
\sum_{i\in B}\frac{c_i}{|D|+1}x_{j,i}+\frac{1}{|D|+1}x_{j,k}-\sum_{i\in B}\frac{c_i}{|D|}x_{j,i} &= \sum_{i\in B}c_i x_{j,i}\left(\frac{1}{|D|+1}-\frac{1}{|D|}\right)+\frac{x_{j,k}}{|D|+1}\\
&= -\left(\sum_{i\in B}\frac{c_i x_{j,i}}{|D|^2+|D|}\right)+\frac{x_{j,k}}{|D|+1}\\
&\le -\left(\frac{m}{|D|+1}\right)+\frac{M}{|D|+1}\\
&\le \frac{M-m}{|D|}\\
\sum_{i\in B}\frac{c_j}{|D|+1}x_{j,i}+\frac{1}{|D|+1}x_{j,k}-\sum_{i\in B}\frac{c_j}{|D|}x_{j,i} &= -\left(\sum_{i\in B}\frac{c_i x_{j,i}}{|D|^2+|D|}\right)+\frac{x_{j,k}}{|D|+1}\\
&\ge -\left(\frac{M}{|D|+1}\right)+\frac{m}{|D|+1}\\
&\ge \frac{m-M}{|D|}
\end{aligned}$$

So we have that the sensitivity is no greater than $\frac{M-m}{|D|}$. □

We will proceed to use this fact to show that Welch's $t$-test will fail to detect this attack.

**Corollary 1.** *Given an audit dataset $S_{audit}$ and significance level $\alpha$, we can use Algorithm 1 to construct a training dataset $S'_{train}$ such that for any feature $j$, $S'_{train}$ passes Welch's t-test when its values in feature $j$ are compared to those of $S_{audit}$ with significance level $\alpha$.*

Before we can prove this corollary, we will need a lemma which bounds the concentration of the Student's $t$-distribution.

**Lemma 2.** *If $X$ and $Z$ are random variables drawn independently from the Student's t-distribution with $\nu$ degrees of freedom and the standard normal distribution respectively, then for every $t>0$, we have*

$$\Pr[|X|<t]\le \Pr[|Z|<t]$$

*Proof.* We will write $F_X(t)$ to denote the CDF of random variable $X$ evaluated at $t$, and $f_X(t)$ the PDF. We will also write $\mathbb{E}_X(g(X))$ to be the expected value of $g(X)$ with randomness over $X$. Let us begin by demonstrating that for all $t<0$, we have $F_X(t)>F_Z(t)$. First, recall that if $W$ and $Y$ are drawn from the $\chi^2$ distribution with $\nu$ degrees of freedom and the standard normal distribution respectively, then $Y\sqrt{\frac{\nu}{W}}$ is distributed according to the Student's $t$-distribution with $\nu$ degrees of freedom, so let us write $X=Y\sqrt{\frac{\nu}{W}}$. Then according to the law of total probability, we have

$$\begin{aligned}
F_X(t) &= \int_0^\infty F_Y\left(t\sqrt{\frac{w}{\nu}}\right)f(w)dw\\
&= \mathbb{E}_W\left(F_Y\left(t\sqrt{\frac{W}{\nu}}\right)\right)
\end{aligned}$$

Notice that $\frac{d^2}{dt^2}F_Y(t)=\frac{d}{dt}f_Y(t)=\frac{d}{dt}\frac{1}{\sqrt{2\pi}}e^{-\frac{t^2}{2}}=-\frac{t}{\sqrt{2\pi}}e^{-\frac{t^2}{2}}>0$ when $t<0$. Then since $t\sqrt{\frac{W}{\nu}}$ must be less than 0, we can apply Jensen's inequality to get

$$\begin{aligned}
F_X(t) &= \mathbb{E}_W\left(F_Y\left(t\sqrt{\frac{W}{\nu}}\right)\right)\\
&\ge F_Y\left(\mathbb{E}_W\left(t\sqrt{\frac{W}{\nu}}\right)\right)\\
&= F_Y\left(t\mathbb{E}_W\left(\sqrt{\frac{W}{\nu}}\right)\right)
\end{aligned}$$

Then since $\frac{d^2}{du^2}\sqrt{u} = -\frac{1}{4\sqrt{u^3}} \leq 0$, we get that $\mathbb{E}_W\left(\sqrt{\frac{W}{\nu}}\right) \leq \sqrt{\frac{\mathbb{E}_W(W)}{\nu}} = \sqrt{\frac{\nu}{\nu}} = 1$. So because $t < 0$, we can see that $t\mathbb{E}_W\left(\sqrt{\frac{W}{\nu}}\right) \geq t$, and since $F_Y(u)$ is increasing, we get

$$\begin{aligned} F_X(t) &\geq F_Y\left(t\mathbb{E}_W\left(\sqrt{\frac{W}{\nu}}\right)\right) \\ &\geq F_Y(t) \end{aligned}$$

Since $f_X$ and $f_Y$ are both symmetric about $t = 0$, it then follows by a symmetric argument that for all $t > 0$, $F_X(t) \leq F_Y(t)$. Then we see that for any $t > 0$,

$$\begin{aligned} \Pr[|X| < t] &= F_X(t) - F_X(-t) \\ &\leq F_Y(t) - F_Y(-t) \\ &= \Pr[|Y| < t] \\ &= \Pr[|Z| < t] \end{aligned}$$

Because $Y$ and $Z$ are independently and identically distributed. □

We are now ready to prove Corollary 1.

*Proof of Corollary 1.* A pair of datasets $D_1, D_2$ pass Welch's $t$-test on feature $j$ if

$$\frac{|\mu_j(D_1) - \mu_j(D_2)|}{\sqrt{\frac{\sigma_1^2}{|D_1|} + \frac{\sigma_2^2}{|D_2|}}} \leq T_{\alpha,\nu}$$

where $\alpha$ is the desired significance level, $\nu$ is the degrees of freedom in the datasets, and $T_{\alpha,\nu}$ is the unique value such that

$$\Pr_{x \sim t(\nu)}[|x| \geq T_{\alpha,\nu}] = \alpha$$

where $t(\nu)$ is the Student's $t$-distribution with $\nu$ degrees of freedom. In our case, the $t$-test compares the reference dataset $S_{audit}$ with the training dataset $S'_{train}$.

The value of $\nu$, and thus the value of $T_{\alpha,\nu}$, depends on the size of the datasets, with the threshold $T_{\alpha,\nu}$ decreasing as the datasets grow large. However, we will use Lemma 2 to give a lower bound for $T_{\alpha,\nu}$ which is constant with respect to $|S'_{train}|$. Then, we will show that by Lemma 1 and Theorem 2 we can use Algorithm 1 to construct a malicious training dataset $S'_{train}$ which maintains an arbitrarily small test statistic, and in particular, a dataset such that the test statistic is below the lower bound on the threshold.

First, let us establish a lower bound on $T_{\alpha,\nu}$. Let us define $T'_\alpha$ to be the unique positive value such that

$$\Pr_{Z \sim \mathcal{N}(0,1)}[|Z| \geq T'_\alpha] = \alpha$$

Then recall that Lemma 2 gives us that

$$\Pr_{X \sim t(\nu)}[|X| < T'_\alpha] \leq \Pr_{Z \sim \mathcal{N}(0,1)}[|Z| < T'_\alpha]$$

If we write $f_X$ and $f_Z$ to represent the probability density functions (PDFs) of $X$ and $Z$ respectively, then we get equivalently that

$$\int_{-T'_\alpha}^{T'_\alpha} f_X(u)du \leq \int_{-T'_\alpha}^{T'_\alpha} f_Z(u)du$$

Then we see that

$$\begin{aligned}
\Pr_{Z\sim\mathcal{N}(0,1)}[|Z| \geq T'_\alpha] &= \Pr_{X\sim t(\nu)}[|X| \geq T_{\alpha,\nu}] \\
\implies \int_{-T'_\alpha}^{T'_\alpha} f_Z(u)du &= \int_{-T_{\alpha,\nu}}^{T_{\alpha,\nu}} f_X(u)du \\
&= \int_{-T_{\alpha,\nu}}^{-T'_\alpha} f_X(u)du + \int_{-T'_\alpha}^{T'_\alpha} f_X(u)du + \int_{T'_\alpha}^{T_{\alpha,\nu}} f_X(u)du \\
&\leq \int_{-T_{\alpha,\nu}}^{-T'_\alpha} f_X(u)du + \int_{-T'_\alpha}^{T'_\alpha} f_Z(u)du + \int_{T'_\alpha}^{T_{\alpha,\nu}} f_X(u)du \\
\implies 0 &\leq \int_{-T_{\alpha,\nu}}^{-T'_\alpha} f_X(u)du + \int_{T'_\alpha}^{T_{\alpha,\nu}} f_X(u)du
\end{aligned}$$

Then because $f_X(x)$ is symmetric about $x = 0$, this yields

$$2\int_{T'_\alpha}^{T_{\alpha,\nu}} f_X(u)du \geq 0$$

and thus

$$\int_{T'_\alpha}^{T_{\alpha,\nu}} f_X(u)du \geq 0$$

Now recall the simple result from calculus that states that if $g$ is positive valued, then

$$\int_a^b g(x)dx \geq 0 \iff a \leq b$$

Then because $f_X$ is positive-valued, our prior result entails that $T_{\alpha,\nu} \geq T'_\alpha$, so $T'_\alpha$ is a lower bound on $T_{\alpha,\nu}$ that does not depend on $|S'_{train}|$.

Next, observe that the test statistic for Welch's $t$-test has the following upper bound:

$$\frac{|\mu_j(S'_{train}) - \mu_j(S_{audit})|}{\sqrt{\frac{\sigma^2_{train}}{|S'_{train}|} + \frac{\sigma^2_{audit}}{|S_{audit}|}}} \leq \frac{|\mu_j(S'_{train}) - \mu_j(S_{audit})|}{\sqrt{\frac{\sigma^2_{audit}}{|S_{audit}|}}}$$

Furthermore, Lemma 1 implies that for any $\varepsilon > 0$, we can choose $\gamma < \frac{\varepsilon}{2}$ such that $\mu_j$ is $(\gamma, c)$-magnitude insensitive, and so by Theorem 2, Algorithm 1 yields a dataset $S'_{train}$ such that $|\mu_j(S'_{train}) - \mu_j(S_{audit})| \leq \varepsilon$ when appropriately parameterized. Then let $\varepsilon = T'_\alpha \frac{\sigma_{audit}}{2\sqrt{|S_{audit}|}}$. This produces the result that

$$\begin{aligned}
\frac{|\mu_j(S'_{train}) - \mu_j(S_{audit})|}{\sqrt{\frac{\sigma^2_{train}}{|S'_{train}|} + \frac{\sigma^2_{audit}}{|S_{audit}|}}} &\leq \frac{2\varepsilon}{\sqrt{\frac{\sigma^2_{audit}}{|S_{audit}|}}} \\
&= \frac{2}{\sqrt{\frac{\sigma^2_{audit}}{|S_{audit}|}}} T'_\alpha \frac{\sigma_{audit}}{2\sqrt{|S_{audit}|}} \\
&= T'_\alpha \\
&\leq T_{\alpha,\nu}
\end{aligned}$$

which passes the $t$-test for feature $j$. Finally, by choosing $k = \max_j \frac{4d(M_j - m_j)\sqrt{|S_{audit}|}}{T'_\alpha \sigma_{audit,j}}$ we get for every feature $i$ that $|\mu_i(S'_{train}) - \mu_i(S_{audit})| \leq 2\min_j T'_\alpha \frac{\sigma_{audit,j}}{2\sqrt{|S_{audit}|}} \leq 2T'_\alpha \frac{\sigma_{audit,i}}{2\sqrt{|S_{audit}|}}$, so $S'_{train}$ passes the $t$-test for feature $i$. □

# D Case Study

We now discuss a number of state of the art works that consider the problem of privacy-preserving auditing. These works are focused on different auditing functions (accuracy, fairness, etc), different

types of machine learning models, and their security models they use are not necessarily aligned. We now briefly outline the techniques and security guarantees that are claimed in each of the works. Our goal is not to provide an exhaustive survey, but rather to illustrate the landscape through recent works that are broadly representative of the field—even though they span different years, venues, and communities (ranging from machine learning to security).

### D.1 Zero Knowledge Proofs for Decision Tree Predictions and Accuracy

**Goal and Solution Details.** Zhang et al. [2020] introduce protocols for auditing accuracy and verifying decision tree predictions. These protocols enable the owner of a decision tree model to prove that the model produces a given prediction on a data sample, or that it obtains a specified accuracy on a given dataset, without revealing any additional information about the model itself. Zhang et al. [2020]'s main contribution is in designing a custom zero-knowledge proof tailored to efficiently verifying the decision tree prediction. The proof consists of algorithms to generate public parameters, custom commitment algorithm for decision tree models, the prover's algorithm which outputs a proof of inference/accuracy, and verifier algorithm to check this proof. The prover, i.e., model provider, must first commit to its model and subsequently demonstrate that the predictions on client queries are consistent with this commitment. For accuracy verification, the authors propose a batching technique that allows to more efficiently checks the correctness of predictions across *multiple* inputs. They then add an extra verification step to determine how many of these predictions match the true labels.

**Security Model.** Zhang et al. [2020]'s security definition is formulated for the case of inference, and follows the traditional zero-knowledge definition structure, which considers two parties (prover and verifier), and where the protocol is required to satisfy correctness, soundness, and zero-knowledge. Either of the two parties can be malicious. In the context of our analysis we are interested in soundness, which specifies whether a malicious prover can deceive the verifier (i.e., auditor), that the prover's hypothesis passed the test. At a high level, the authors' soundness definition can be summarized as follows: A prover should not be able to output a commitment to a tree $\mathcal{T}$ along with a proof $\pi$, prediction $y$ and datapoint $a$ such that the verifier accepts the proof and at the same time, the $\mathcal{T}'s$ prediction for $a$ is not equal to $y$. Definition of soundness for the accuracy case is similar: the prover outputs the dataset which is used for checking accuracy, and wins the game if the verifier accepts the proof even though the accuracy is not what the prover claims it to be.

**Discussion.** The security notion in this work aligns well with the intuitive goals of verifying both the correctness of individual predictions and the accuracy of a model on a given dataset. However, it does not give any formal guarantees for datasets beyond the audited dataset, i.e., the accuracy verification solution does not generalize to other datasets drawn from the same distribution. In fact, Zhang et al. [2020] explicitly note that it is possible to use their solution to check accuracy on a *public dataset*. In this setting, their approach falls within our framework of Definition 3, and is vulnerable to the same attack as outlined in §4.1. In fact, note that our example works even given an *ideal* proof of accuracy (when it is checked on a dataset known to the adversary), and even if the prover supplies an *additional* proof of training to complement its proof of accuracy.

### D.2 P2NIA: Privacy-Preserving Non-Iterative Auditing

**Goal and Solution Details.** Bourrée et al. [2025] propose a novel auditing scheme that enables one-shot verification of a model's group fairness while preserving privacy for both parties: the model provider is not required to open-source the model, and the auditor need not disclose any private information to support the audit. The main contribution of Bourrée et al. [2025] is a mechanism that enables auditing without requiring the auditor to supply the audit dataset. Specifically, the model provider supplies a dataset together with the corresponding predictions (both in the clear), which the auditor then uses to verify the fairness condition. To construct this dataset, model provider draws on a portion of its internal training data. To preserve confidentiality of this data, it is not shared directly. Instead, model provider feeds it into a synthetic data generation algorithm, and the resulting synthetic dataset is what is sent to the auditor.

**Security Model.** The work does not provide a formal security model. It is set up in the black-box setting and assumes that the auditor does not know the distribution of the model owner's training data.

**Discussion.** As Bourrée et al. [2025] do not utilize cryptographic techniques to prove that the outputs actually correspond to the given inputs, the prover can easily cheat by simply adjusting the labels it supplies for the constructed dataset. However, even if one were to strengthen the scheme by adding a secure proof of training (e.g., Pappas and Papadopoulos [2024]) together with inference proofs (as in Zhang et al. [2020]), the fact that the model owner knows the dataset that is being used for the audit means that the solution falls within our framework of Definition 3, and is thus vulnerable to data-forging attacks. An interesting open question would be to see if, since in this scenario the model owner not only knows, but directly influences the audit dataset, there can be an even simpler attack.

### D.3 Confidential-PROFITT: Confidential PROof of FaIr Training of Trees

**Goal and Solution Details.** Shamsabadi et al. [2022] propose Confidential-PROFITT, a framework for certifying fairness of decision trees while preserving confidentiality of both the model and the training data. Confidential-PROFITT consists of a zero-knowledge-friendly decision tree learning algorithm that, when executed honestly, enforces fairness by design—up to a tunable degree controlled by a parameter. On top of this, Confidential-PROFITT designs a zero-knowledge proof system to verify fairness of a decision tree. The proof requires the model provider to commit to both the model and its training data, then prove in zero-knowledge that the paths taken by the committed training points through the (committed) decision tree satisfy specified fairness bounds. In terms of fairness metrics, Confidential-PROFITT supports *demographic parity* and *equalized odds* as fairness metrics.

**Security Model.** Confidential-PROFITT considers a malicious model provider (that, however, is assumed to commit to the training data honestly) and a malicious auditor (who wishes to learn model details/training data), and obtains standard zero-knowledge proof properties (correctness, soundness, zero-knowledge) with respect to a statement that can be summarized roughly as follows "With respect to a private dataset *chosen by the model provider*, the committed model satisfies certain fairness guarantees".

**Discussion.** Confidential-PROFITT assumes that the model provider honestly commits to the training data. Under this assumption, the corresponding zero-knowledge proof certifies that the resulting model inherits the fairness guarantees of the fair learning algorithm introduced in Confidential-PROFITT (which the authors show indeed improves fairness). However, if the provider is not restricted to committing to the true training data, Confidential-PROFITT is vulnerable to data-forging attacks, as the provider can choose the audit dataset before committing to the model.

### D.4 OATH: Efficient and Flexible Zero-Knowledge Proofs of End-to-End ML Fairness

**Goal and Solution Details.** Franzese et al. [2024] present OATH, a model-agnostic fairness auditing framework. The core idea in OATH is to leverage clients (who query the model during deployment) to participate in the auditing process. OATH operates in two phases: (i) a certification protocol between the model provider and the auditor, and (ii) a query authentication protocol involving model provider, inference clients, and auditor (dubbed verifier in OATH). The first phase follows the standard certification flow we describe in §3. In the second phase, the auditor receives commitments to client queries and the corresponding model predictions. These commitments can later be verified in zero knowledge for fairness, correctness, and consistency with the certified model.

**Security Model.** OATH considers three fully malicious entities: a model provider, inference clients, and an auditor. These parties are assumed not to collude with each other. The auditor assesses model fairness both with respect to the calibration dataset and the clients queries. The system provides standard correctness, soundness, and zero-knowledge with respect to these two datasets.

**Discussion.** The calibration dataset which is used in the certification protocol between the model provider and the auditor might be supplied by either party. If the calibration dataset is chosen by the prover, same as P2NIA and Confidential-PROFITT, the corresponding fairness check is vulnerable to data forging. However, in contrast to prior works, OATH can fall back on guarantees based on client's queries.

### D.5 FairProof: Confidential and Certifiable Fairness for Neural Networks

**Goal and Solution Details.** Yadav et al. [2024] propose FairProof, a fairness certification approach that maintains confidentiality of the model. In contrast to Confidential-PROFITT and OATH, which

focus on group fairness metrics, FairProof considers local individual fairness. This allows Yadav et al. [2024] to issue a personalized certificate to every client.

**Security Model.** FairProof system involves a malicious model provider and malicious clients (who wish to learn model details/training data), and considers standard correctness, soundness, and zero-knowledge properties. The corresponding statement is roughly as follows: "Given a datapoint $x$, the model's output is $y$ and a lower bound on an individual fairness parameter for $x$ is $\epsilon_x$".

**Discussion.** The usage of a specific fairness metric (local individual fairness) allows FairProof to provide per-client certificates of fairness, and escape the problems that arise from the usage of reference datasets (including vulnerability to data-forging attacks). On the flip side, FairProof requires to generate fairness certificates during deployment and does not provide any fairness guarantees prior to deployment.

### D.6 zkCNN: Zero Knowledge Proofs for Convolutional Neural Network Predictions and Accuracy

**Goal and Solution Details.** Liu et al. [2021] propose zkCNN, a zero-knowledge proof protocol for inference and accuracy of convolutional neural networks (CNNs). The core contribution is a novel sumcheck protocol (which is the key ingredient in many zero-knowledge system) that is tailored to two-dimensional convolutions.

**Security Model.** zkCNN considers the standard setting with a prover and a verifier. Either party can be malicious. Liu et al. [2021]'s security definition for inference is a zero-knowledge-style definition, and the scheme is required to satisfy correctness, soundness, and zero-knowledge. Similar to Zhang et al. [2020], Liu et al. [2021]'s soundness intuitively states that a prover should not be able to output a commitment to a model and provide a proof $\pi$, prediction $y$ and datapoint $X$ such that the verifier accepts the proof, and at the same time, the committed model's prediction for $X$ is not equal to $y$. If instantiated with a specific commitment scheme, Liu et al. [2021]'s scheme further satisfies knowledge soundness, the stronger version of soundness where there exists an extractor to extract the CNN parameters from a valid proof and prediction with overwhelming probability. Liu et al. [2021] do not provide a security definition for their proof of accuracy.

**Discussion.** As Liu et al. [2021] do not give a security definition for their proof of accuracy, the formal security guarantee they provide is not fully clear. However, the authors indicate that their scheme can be used to prove the accuracy on a public dataset. This scenario falls within our framework of definition 3, and is vulnerable to the same style of attack as outlined in §4.1.

### D.7 Scaling up Trustless DNN Inference with Zero-Knowledge Proofs

**Goal and Solution Details.** Kang et al. [2022] propose a zero-knowledge-based framework for verifying DNN inference and accuracy. Their key contribution is a careful translation of DNN specifications into arithmetic circuits suitable for zero-knowledge proofs. The system also introduces economic incentives to support ML-as-a-service. Concretely, when verifying accuracy, the model provider first commits to the model, and the client commits to the test set. Both parties then deposit monetary collateral into an escrow. The client reveals the test set, and the provider must produce a zero-knowledge proof that the committed model meets the claimed accuracy. If the provider fails or refuses to prove the required accuracy, it forfeits its collateral; otherwise, the client pays for the service.

**Security Model.** Kang et al. [2022] study the standard two-party setting with a *prover* (model provider) and a *verifier* (client), either of whom may be malicious. Cryptographically, they aim for the standard zero-knowledge proof properties: *completeness*, *knowledge soundness*, and *zero knowledge*. They further consider incentives, showing that—under certain assumptions—honest model providers and clients are motivated to participate in the accuracy verification protocol, while malicious parties are discouraged.

**Discussion.** In terms of cryptographic guarantees, Kang et al. [2022] gets the core design right: their protocol for proofs of accuracy closely follows the framework where the model provider commits to the model before the audit and is not vulnerable to our data-forging attacks. However, Kang et al. [2022] provide no formal guarantees about accuracy on data outside the audited set. It would be interesting to perform a formal analysis in a suitable security model given their constraints.

### D.8 ezDPS: An Efficient and Zero-Knowledge Machine Learning Inference Pipeline

**Goal and Solution Details.** Wang and Hoang [2023] introduce ezDPS, a pipeline for zero-knowledge proofs of inference correctness and accuracy above a specified threshold. They construct arithmetic circuit gadgets for key ML operations, including exponentiation, absolute value, and array max/min, and further devise optimized methods for proving Discrete Wavelet Transform, Principal Component Analysis, and multi-class Support Vector Machines with various kernel functions using an efficient set of arithmetic constraints.

**Security Model.** Wang and Hoang [2023] consider two mutually distrusting parties – a malicious server and a semi-honest client, who follows the protocol but aims to learn information about the model's parameters. For their inference pipeline, they consider standard definitions of correctness, soundness, and zero-knowledge (similar to those by Zhang et al. [2020] and Kang et al. [2022]). Wang and Hoang [2023] do not provide a security definition for their proof of accuracy.

**Discussion.** Similar to Liu et al. [2021], as Wang and Hoang [2023] do not provide a security definition for their proof of accuracy, the precise security guarantee they achieve is somewhat unclear. However, Wang and Hoang [2023] indicate that their scheme can be used to prove the accuracy on a public dataset, which falls within our framework of definition 3. This instantiation of their method is vulnerable to the same style of attack as outlined in §4.1.

### D.9 Confidential-DPproof: Confidential Proof Of Differentially Private Training

**Goal and Solution Details.**Shamsabadi et al. [2024] present Confidential-DPproof, a framework that enables the model provider to prove to an auditor that their model was correctly trained via DP-SGD, a classic approach for training models with differential privacy guarantees. The certification of DP-SGD's training run is done in zero-knowledge.

**Security Model.** Shamsabadi et al. [2024] consider two mutually distrusting parties: a prover, i.e., model provider, and an auditor. The prover is fully malicious, while the auditor is semi-honest and aims to obtain information about the model's parameters. Confidential-DPproof considers standard definitions of correctness, soundness, and zero-knowledge.

**Discussion.** The data used by Shamsabadi et al. [2024] for their zero-knowledge proof is selected by the prover. This fits the framework in §3, and makes the solution susceptible to data-forging attacks. In particular, a malicious prover could degrade the claimed differential privacy guarantees by, for example, supplying multiple copies of its (otherwise honest) training data as the input to the Confidential-DPproof protocol. We leave a formal treatment and full development of this attack as an interesting direction for future work.

## E Related Work (Continued)

A number of recent works aim to prove desirable model properties. In terms of *what* these works prove, they can be roughly categorized into proofs of training, inference, accuracy, and fairness. In terms of *how* the corresponding protocols work, recent works on certifiable ML can be categorized as follows:

**Cryptographic approaches** A prolific line of research adapts various cryptographic techniques to certify properties such as accuracy, fairness, etc., without revealing the model's details. The most common technique is *zero-knowledge proofs* (zk proofs), which allow to formally prove that a model satisfies certain properties without revealing anything else about the model. They have been used to certify fairness [Shamsabadi et al., 2022, Yadav et al., 2024, Franzese et al., 2024, Zhang et al., 2025b], inference [Zhang et al., 2020], accuracy [Zhang et al., 2020], and to prove that the model has been trained using a certain algorithm [Abbaszadeh et al., 2024, Garg et al., 2023, Sun et al., 2024, Pappas and Papadopoulos, 2024] (without revealing the training data). Other works [Duddu et al., 2024, Chang et al., 2023] use *secure multi-party computation* (MPC), which allows mutually distrusting parties to jointly compute on private inputs without revealing anything about the inputs apart from the outcome.

**Black box auditing/Statistical testing** These approaches probe a model by submitting inputs, collecting outputs, and analyzing them for undesirable behavior. Tramer et al. [2017], Saleiro et al.

[2018] use black-box testing to check for potential unfairness or bias, while [Tan et al., 2018] distill a new model to gain insight into the black box one.

**Outside-the-box auditing** Here the model owner provides access to information beyond query responses, such as source code, documentation [Mitchell et al., 2019], hyperparameters, training data, deployment details, or internal evaluation results.

Finally, we note that our work is related to, but distinct from, data poisoning attacks. We discuss the relationship between the two works below.

# F Cryptographic Auditing of ML: Background and Subtleties

We outline different categories of proofs that are used in the context of auditing machine learning algorithms. For simplicity, from now on we assume that the *training algorithm is public* (note that making it private only makes the adversary in our attacks stronger, i.e., it could potentially be *easier* for the model owner to perform a data-forging or any other type of attack).

**Proof of Training** A *proof of training* can be viewed as a zero knowledge proof for the following relation $\mathcal{R}$: given $\mathsf{x} = (\mathsf{com}_h, \mathsf{com}_S)$, and $\mathsf{w} = (h, S_{train}, \rho, \rho_h, \rho_S)$, $\mathcal{R}$ outputs 1 if and only if $\mathsf{Train}(S_{train}; \rho) = h$, $\mathsf{com}_h = \mathsf{Commit}(h; \rho_h)$ and $\mathsf{com}_S = \mathsf{Commit}(S_{train}; \rho_S)$, where $\rho$ is the randomness used for training. Here, Commit is a commitment scheme (§A.5). Intuitively, here the commitment lets the prover fix $h$ and $S_{train}$ up front without revealing them.

**Proof of Inference** A *proof of inference* can be viewed as a special case of zero knowledge proof for the following relation $\mathcal{R}$: given $\mathsf{x} = (\mathsf{com}, x, y)$, and $\mathsf{w} = (h, \rho_h)$, $\mathcal{R}$ outputs 1 if and only if $h(x) = y$ and $\mathsf{com} = \mathsf{Commit}(h; \rho_h)$.

**Auditing using Zero Knowledge Proofs** The strongest form of ZK-based auditing arises when the prover first produces a *proof of training*, thereby showing that a specific committed model instance came from an honest training procedure on a private dataset, and subsequently provides a *proof of property* attesting that the committed model meets the desired criterion. Let $f$ be an auditing function outputting a binary that takes as input a training data set $S_{train}$, an auditing data set $S_{audit}$, and the model $h$. Then privacy-preserving auditing can be realized using zero knowledge proofs for the following relation $\mathcal{R}$: given, $\mathsf{x} = (\mathsf{com}_h, \mathsf{com}_S, S_{audit})$, and $\mathsf{w} = (h, S_{train}, \rho, \rho_h, \rho_S)$, $\mathcal{R}$ outputs 1 if and only if $\mathsf{Train}(S_{train}; \rho) = h$, $f(S_{train}, S_{audit}, h) = 1$, $\mathsf{com}_h = \mathsf{Commit}(h; \rho_h)$ and $\mathsf{com}_S = \mathsf{Commit}(S_{train}; \rho_S)$.

**Definition Subtleties** The zero knowledge property ensures confidentiality of the committed model and training data. However, as we shall see next, knowledge soundness does *not* necessarily capture the actual goal of the auditing process. The reason is that knowledge soundness is typically defined with respect to statements $\mathsf{x} = (\mathsf{com}_h, \mathsf{com}_S, S_{audit})$, which **(1)** are bound to a specific dataset $S_{audit}$, and **(2)** do not specify how or when each component of $\mathsf{x}$ is generated. In practice, it is plausible that $S_{audit}$ is supplied by verifier (i.e., the auditor). We show that if a cheating prover (i.e., model owner) adaptively generates $\mathsf{com}_{h^*}$ and $\mathsf{com}_{S^*}$ *after* observing $S_{audit}$, it is possible to pass the zero knowledge auditing process after maliciously crafting model $h^*$ and/or training data $S^*$. Furthermore, we show that $h^*$ behaves pathologically when evaluated on data outside $S_{audit}$, in a way that completely undermines the purpose of the auditing process.

We note that while this subtlety was indeed overlooked in several works on zero-knowledge-based auditing, it applies even more directly to various non-cryptographic auditing approaches that do not enforce a secure commitment from the prover.

# G Further Evaluation

First, we present in Figure 4 results for attacking accuracy audits on additional datasets mentioned in §5.

Next, we present the application of the attack described in § 4.1. In this attack, the adversary is attempting to maximize the model's denial rate $\Pr_{x\sim\mathcal{D}}[h(x) = 0]$ while still appearing accurate to the audit. The results of this attack are given in Figure 5. Observe that as the attack parameter approaches 1 (and the attack becomes maximally malicious), the denial rate of the model on the audit set remains close to the fully honest denial rate while the denial rate on independently sampled

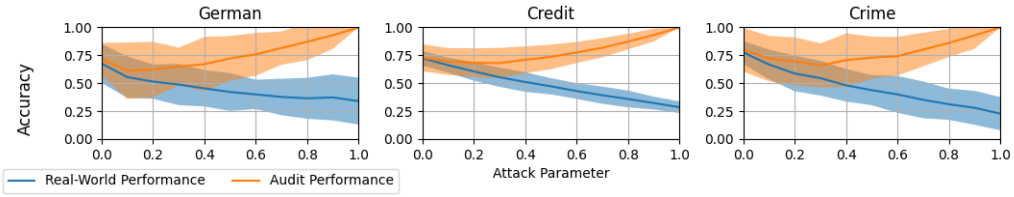


Figure 4: Accuracy of models trained on datasets constructed by Algorithm 1 on various benchmarks. Values are averages over ten runs, error bars represent one standard deviation.

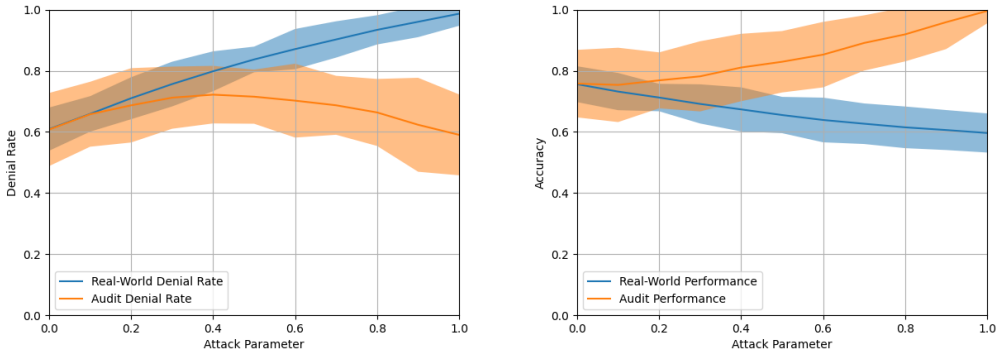


Figure 5: Accuracy and denial rates of models trained on datasets constructed by Algorithm 1 on ACSEmployment. Values are averages over ten runs, error bars represent one standard deviation.

data approaches $1$. Similarly, the accuracy of the model on the audit set approaches $1$, while the true accuracy decreases down to roughly $0.6$ (this reflects the true denial rate of the distribution).

Next we examine our ability to observe these attacks by applying statistical tests to the datasets, as described in § C. There is no singular way to determine whether two samples were drawn from the same distribution, so we apply some common statistical tools. In particular, our goal is to determine if the distribution from which the audit data is drawn is identical to the distribution from which the training data is drawn. We use Welch's $t$-test, which serves to determine whether two distributions have the same mean, and Levene's test, a one-way ANOVA for determining whether two distributions have the same variance. These tests are typically applied to 1-dimensional data, and so we apply them to each feature individually. The results of these experiments are given in Table 2.

We observe that the summary statistics of the malicious training data closely match the values for the honest data, suggesting that comparing these two values would not be a successful detection mechanism. This is compounded by the fact that the test statistics for Welch's $t$-test and Levene's test for the malicious training data are considerably smaller on average than the same test statistics for the honest training data, corroborating higher rate of passing the hypothesis tests we observe. At a significance level of $\alpha = 0.05$, we expect a false positive rate of approximately $5\%$. On the other hand, we observe a $0\%$ true positive rate. We note that in a practical application of this attack, the auditor would have access only to the honest or malicious values over a single training run, and would thus be unable to easily distinguish between the two cases by comparing the values or by looking at averages over many runs as we have done here. That being said, an auditor may find it suspicious if the p-value returned by a statistical test is extremely low (even though such a scenario may be very plausible for some distributions); an attacker can safely relax this attack to a comfortable degree, though doing so will increase the risk of failing the audit.

Finally, we present an evaluation of a modified version of the attack that targets neural networks rather than decision trees. Whereas decision trees have very specific conditions that allow us to constrain their behavior, it is much harder to provide theoretical guarantees for neural networks. In

Table 2: Summary and Test statistics for Age feature on ACSEmployment, conditioned on label. Test statistics used are Welch's $t$-test and Levene's test. Attack is undetectable when summary statistics are similar to honest ones, and when test statistics are close to 0. Comparisons are between fully honest and fully malicious datasets.

| Age | | Label = 0 | | Label = 1 | |
|---|---|---|---|---|---|
| | | Honest | Attack | Honest | Attack |
| Summary Statistics | $\mu$ | 41.6651 | 41.9657 | 43.9184 | 43.8131 |
| | $\sigma^2$ | 804.5804 | 810.8822 | 223.1269 | 221.42394 |
| Test Statistics | $t$-test | 0.6521 | 0.0033 | 0.7067 | 0.0110 |
| | ANOVA | 0.6200 | 0.0026 | 1.6500 | 0.0186 |

| Education | | Label = 0 | | Label = 1 | |
|---|---|---|---|---|---|
| | | Honest | Attack | Honest | Attack |
| Summary Statistics | $\mu$ | 13.39761692 | 13.41700338 | 18.45539675 | 18.50545506 |
| | $\sigma^2$ | 42.99789908 | 42.16899485 | 9.979327135 | 8.943082831 |
| Test Statistics | $t$-test | 0.7984001575 | 0.0356390553 | 0.9499974697 | 0.1302788154 |
| | ANOVA | 0.4844657261 | 0.0003374130653 | 1.227829625 | 0.02531893152 |

| Military Status | | Label = 0 | | Label = 1 | |
|---|---|---|---|---|---|
| | | Honest | Attack | Honest | Attack |
| Summary Statistics | $\mu$ | 2.5794 | 2.5834 | 3.8121 | 3.8302 |
| | $\sigma^2$ | 3.2749 | 3.2648 | 0.3507 | 0.3265 |
| Test Statistics | $t$-test | 0.4997 | 0.0313 | 0.8699 | 0.1755 |
| | ANOVA | 1.0240 | 0.0009 | 1.2394 | 0.0304 |

order to encourage memorization of the training data, we used a relatively shallow network with very large individual layers. Our attack samples a large amount of training data, and decides whether to label each point with the honest label or dishonest label depending on its proximity to the nearest audit data point. We evaluated this attack on an 8-dimensional mixture of Gaussian distributions; the results are shown in Figure 6.

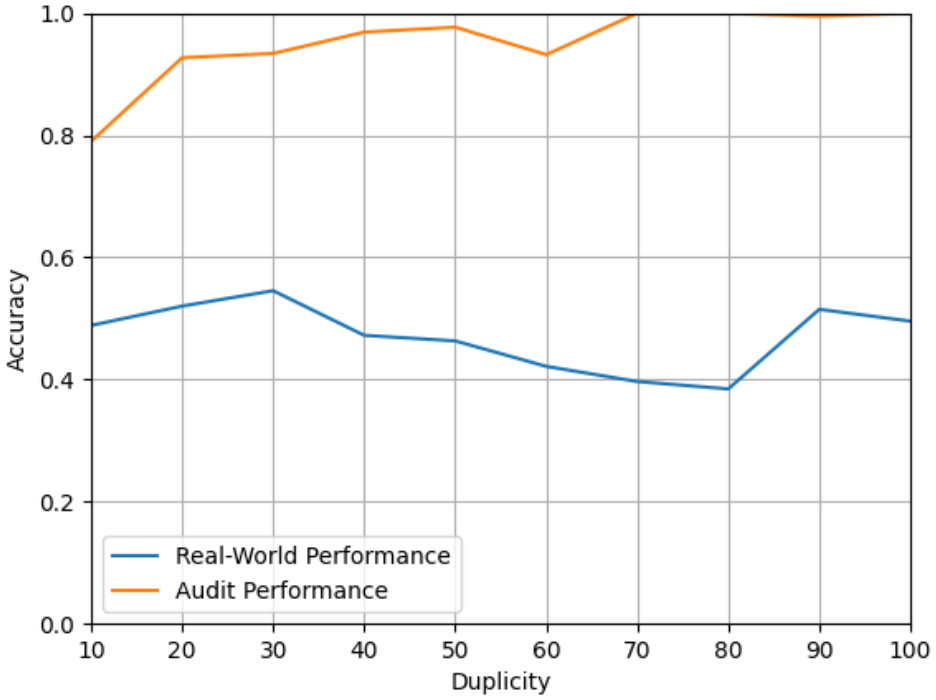


Figure 6: Performance of 226M-parameter neural networks trained on datasets constructed from a mixture of Gaussian distributions. Duplicity refers to the number of perturbed copies of the audit dataset included in the training data.

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our main technical results are in §4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We elaborate on limitations of the undetectability of our attacks in §C.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We give a set of proofs in §B.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We explain how we performed the experiments in §5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We will provide open access to the data and code in the future.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We explain how we performed the experiments in §5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We explain how we performed the experiments in §5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

Justification: The authors used LLMs to polish writing, and for assistance with literature search in some components of this paper. We also used generative AI to create some of the icons in Figure 1. In addition, we used an LLM for assistance with Lemma 2. We checked the proof assistance thoroughly by hand before including it in the paper.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.