Clean-Label Poisoning for Gradient-Boosted Decision Trees

Sinuo Fan
Tufts University
Medford, MA, USA
sinuo.fan@tufts.edu

Jun Woo Chung
Rochester Institute of Technology
Rochester, NY, USA
jc4303@rit.edu

Weijie Zhao
Rochester Institute of Technology
Rochester, NY, USA
wjz@cs.rit.edu

Yingjie Lao
Tufts University
Medford, MA, USA
yingjie.lao@tufts.edu

Abstract

Clean-label poisoning attacks have been well studied for differentiable models, yet their practical behavior in gradient-boosted decision trees (GBDTs) remains less understood. In this paper, we investigate clean-label poisoning under realistic constraints: class labels are preserved, and features after perturbation must remain plausible. Our framework selects poisoning candidates using a tree-specific influence score and perturbs input features that prioritize dimensions based on split-gain signals. A key finding is a non-monotonic dependence on the perturbation budget ε, arising from discrete threshold crossings in tree splits and feasibility constraints on the perturbation. We introduce an ellipsoidal region and project both update iterates and finite-difference probes onto this region to obtain meaningful and feasible perturbations. Experimental results show that our attack is highly effective, e.g., F1 score on Adult dataset is 0.71 to 0.43, and F1 score 0.54 to 0.33 on Credit-g.

1 Introduction

Modern decision systems increasingly rely on machine learning models trained on large, continuously updated datasets. This expands the attack surface of the training pipeline: an adversary who can modify a small portion of training data may steer the learned model in harmful ways, even when inputs are untouched during inference. Among training-time threats, clean-label poisoning is particularly challenging yet powerful: the attacker must preserve labels and ensure that modified samples remain plausible, thereby making the attack more stealthy and harder to detect.

Clean-label poisoning has been well studied for differentiable models, where gradients naturally guide perturbations [Shafahi et al., 2018, Aghakhani et al., 2021, Huang et al., 2020]. In contrast, gradient-boosted decision trees (GBDTs)—a dominant choice for tabular learning—have distinct properties: predictions are *piecewise constant* and change mainly when perturbations cross discrete split thresholds. Thus, existing clean-label poisoning attacks on deep neural networks (DNNs) cannot be directly transferred to GBDT. Despite widespread deployment of GBDT in security- and risk-sensitive settings, its clean-label poisoning behavior under realistic constraints remains underexplored.

In this paper, we address this gap by introducing a novel clean-label poisoning attack against GBDT. The overview of the method is shown in Figure 1. In this threat model, the adversary perturbs a limited number of feature vectors while preserving their ground-truth labels (i.e., the attacker may modify only features but cannot alter the dataset labels). In addition, to ensure the poisoned samples

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

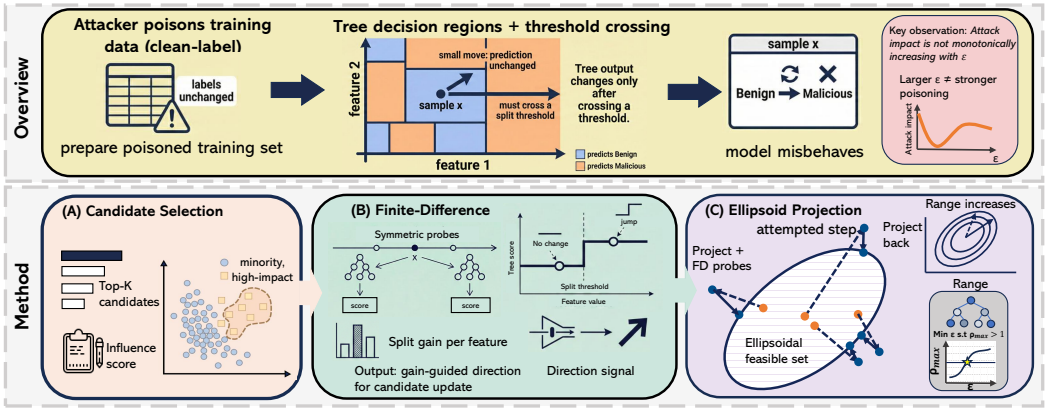

Figure 1: **Method Overview.** *Top:* Key observation: larger perturbation budgets do not guarantee stronger attacks due to threshold-crossing dynamics. *Bottom:* Our approach: (A) select high-influence candidates; (B) estimate ascent directions via finite-difference; (C) project to an ellipsoid constraint to ensure plausibility.

remain realistic, we constrain perturbations to follow the data distribution rather than bounding each feature independently.

A key empirical finding is that increasing the perturbation budget ε does not necessarily lead to stronger poisoning effects. As shown in the top right corner of Figure 1, increasing ε does not consistently strengthen the attack: performance degradation and attack success often improve initially, then saturate or decline. This is different from observations of attacks against DNNs, where larger perturbation budgets typically lead to stronger and more reliable poisoning effects due to smooth loss landscapes and continuous parameter updates [Kantchelian et al., 2016, Calzavara et al., 2025]. In contrast, the discrete split thresholds and path-dependent behavior of tree ensembles yield different training dynamics from DNNs, leading to larger perturbations that can diminish or even be detrimental once feasibility constraints and leaf reassignment saturation dominate. Large perturbations can push samples into split regions, thereby reducing the impact of poisoning. Hence, to enhance the attack performance, we introduce an *ellipsoidal region* learned from the candidate pool. We project both update iterates and finite-difference probe points into the ellipsoid, yielding a perturbation boundary that aligns the benign data distribution. By projecting back onto the ellipsoid, we cap the effective perturbation, even when the budget is large, preventing the attack from drifting into unfavorable regions.

The contributions of this paper are summarized below:

(i) We empirically characterize non-monotonic sensitivity of clean-label poisoning attack against GBDTs with respect to ε;

(ii) We propose a novel methodology that perturbs the input with an ellipsoidal region to bound the perturbation to achieve better attack performance;

(iii) Through comprehensive evaluation across multiple benchmark tabular datasets, we demonstrate that our method achieves model performance degradation while maintaining remarkable detection resistance against supervised classifiers and preserving the training data distribution.

2 Background and Related Work

2.1 GBDT: Discrete Structure and Implications

Gradient-boosted decision trees (GBDTs) are a standard choice for tabular learning [Il'ev, 2001, Chen and Guestrin, 2016, Prokhorenkova et al., 2018, Grinsztajn et al., 2022]. A trained ensemble predicts by routing each sample through a sequence of threshold tests, and the final leaf value determines the output [Yang, 2021]. Because routing depends on discrete split thresholds, the prediction function of a fixed ensemble is piecewise constant over large regions of the feature space, and it changes only when a perturbation crosses one or more thresholds and sends the sample to a different leaf

[Andriushchenko and Hein, 2019, Einziger et al., 2019]. This discrete behavior differs from smooth, gradient-based models. Small perturbations often leave the routing path unchanged, so the prediction remains the same. Larger perturbations can change the routing decisions, causing an abrupt change in the leaf assignment and the predicted value [Chen et al., 2017]. Unlike neural networks that optimize over continuous parameter spaces, GBDTs construct trees sequentially by greedily selecting splits that maximize information gain or minimize loss [FRIEDMAN, 2001]. Each tree is trained on residuals from the previous ensemble, and the final prediction is a weighted sum of leaf outputs. This sequential structure makes GBDTs highly effective on tabular data, consistently outperforming deep learning on structured datasets with limited samples or strong feature heterogeneity [Grinsztajn et al., 2022]. These sequential dependencies also motivate tree-specific data updates, as Lin et al. [2023] retain auxiliary statistics and selectively update subtrees to unlearn training records in GBDTs.

2.2 Clean-Label Poisoning Attacks

Data poisoning attacks modify a small portion of the training set to influence the model learned from it [Barreno et al., 2010]. Compared to *dirty-label* attacks where the adversary can modify the class labels, *clean-label* poisoning is more stringent, as the adversary can only influence the input data, which makes the poisoned data harder [Steinhardt et al., 2017]. Most established clean-label attacks focus on differentiable models such as DNNs, where gradients provide a direct signal for choosing how to perturb training inputs [Biggio et al., 2012]. A common strategy is to modify inputs so they remain consistent with their labels, while steering training toward harmful behavior through gradient-based objectives [Shafahi et al., 2018, Zhu et al., 2019]. These ideas do not transfer directly to GBDTs. For tree ensembles, small input changes often leave split decisions unchanged, and prediction changes are driven mainly by threshold crossings and leaf routing rather than smooth gradient information [Huang et al., 2020]. Clean-label attacks can be further categorized into *targeted* attacks, which aim to misclassify specific test samples [Aghakhani et al., 2021], and *availability* attacks, which degrade overall model performance without a specific target [Fowl et al., 2021, Lu et al., 2023]. For example, Shafahi et al. [Shafahi et al., 2018] craft poisoned samples by optimizing feature perturbations to maximize the loss on a target test input, while recent work explores unlearnable examples that make training data unusable [Huang et al., 2021]. The stealthiness of clean-label poisoning makes it particularly challenging to detect, as poisoned samples retain their original labels and may appear indistinguishable from benign data under standard statistical tests. CLPA uses a two-phase generative adversarial framework to construct plausible clean-label availability poisons [Zhao and Lao, 2022]. It synthesizes samples with a distribution similar to naturally misclassified inputs while preserving their labels. This supports an availability attack that reduces overall model accuracy while keeping the poisoned samples visually plausible.

2.3 Attacks against GBDT: Evasion, Poisoning, and Supporting Tools

Most prior work on adversarial attacks against tree models has focused on evasion attacks during inference [Zhang et al., 2020, Devos et al., 2021]. Kantchelian et al. [2016] formulate optimal evasion for tree ensembles via mixed-integer linear programming (MILP) and propose adversarial boosting as a hardening strategy. Chen et al. [2019] develop efficient robustness verification algorithms for gradient boosted decision trees, reformulating the problem as max-clique on multi-partite graphs. Calzavara et al. [2021] propose feature-partitioned forests with certified robustness guarantees against bounded perturbations. Recent work by Cascioli et al. [2024] accelerates repeated adversarial example generation by exploiting feature-level regularities across sequential attack problems. In contrast, poisoning attacks against tree models remain less studied. Recently, *Timber* [Calzavara et al., 2025] presents a white-box poisoning approach based on greedy search with efficient subtree retraining.

Among these prior works, data attribution methods provide a natural basis for prioritizing candidate points under a limited poisoning budget. Influence functions [Koh and Liang, 2017] and TracIn [Pruthi et al., 2020] estimate training data sample impact in differentiable models, while tree-specific approximations such as LeafRefit [Sharchilev et al., 2018] and BoostIn [Brophy et al., 2023] adapt these ideas to ensemble trees by exploiting leaf statistics and boosting dynamics. When analytic gradients are unavailable or unreliable, as is the case for tree models, finite-difference methods offer a practical alternative for estimating local sensitivity and guiding optimization [Chen et al., 2017, Madry et al., 2018]. For tabular data, however, constraining each feature independently is often insufficient to ensure semantic or statistical plausibility, since real-world features exhibit strong correlations. Prior work has used Mahalanobis-style geometry to quantify how far a sample deviates from the training

distribution and to identify out-of-distribution inputs [Lee et al., 2018]. These methods motivate the design of our attack. Related studies address model integrity, ownership, and efficiency [Fan and Lao, 2026]. Fragile signatures in selected tree leaves support integrity authentication [Zhao et al., 2022], while in-place fine-tuning embeds robust watermarks in GBDTs [Chung et al., 2026].

3 Methodology

3.1 Problem Formulation and Threat Model

Setup. Consistent with prior works [Oprea et al., 2022, Chaalan et al., 2024, Zheng and Li, 2021], we study classification tasks on tabular data. The clean training set D contains N samples, each consisting of a d-dimensional feature vector $\mathbf{x}$ and a label y. A gradient boosting decision tree (GBDT) model F is trained on D to predict labels from features. Model performance is evaluated on a held-out test set using standard metrics such as accuracy, AUC, F1 score, and recall.

Threat model. The attacker perturbs the input features of a small subset of training data while keeping their labels unchanged. Let $\rho \in (0, 1]$ denote the fraction of training data that can be poisoned, and let $K = \lfloor \rho N \rfloor$ be the number of poisoned samples. For each selected sample $(\mathbf{x}, y)$, the attacker computes a perturbation $\mathbf{r}$ to obtain $\mathbf{x}' = \mathbf{x} + \mathbf{r}$.

The attacker does not have access to the victim model. Instead, a surrogate GBDT F_s is trained on the clean training set using the same model class. The surrogate model helps define the loss that the attacker aims to increase, while providing information used to prioritize features in finite-difference estimation.

Perturbation magnitude. To ensure poisoned samples remain imperceptible, we bound perturbations relative to each feature's natural variability. Let $\boldsymbol{\sigma}$ denote the feature standard deviations. The perturbation satisfies $\mathbf{r} = \varepsilon \cdot \boldsymbol{\sigma}$, where $\varepsilon > 0$ controls the attack strength and the inequality holds element-wise.

3.2 Finite-Difference Estimation

A key challenge in poisoning GBDTs is that the model output is piecewise constant with respect to input features, so the analytical gradient is zero almost everywhere. Standard gradient-based updates rely on local gradients to choose a direction, but for GBDTs the loss often does not change under small input steps, making such updates ineffective. To address this, we estimate directions using finite-difference (FD) on the surrogate model.

Recall from Section 3.1 that the attacker trains a surrogate GBDT on the clean training set D. Let F_s denote this surrogate model and L denote the loss function (e.g., cross-entropy). For a selected sample $(\mathbf{x}, y)$, the attacker aims to increase the surrogate loss $L(F_s(\mathbf{x}'), y)$ by perturbing $\mathbf{x}$ to $\mathbf{x}'$.

We estimate the direction coordinate-wise using centered finite-difference. Let $\mathbf{W}$ denote a feature weight vector extracted from F_s. For coordinate j, we form two probes $\mathbf{x}^+$ and $\mathbf{x}^-$ by changing only the j-th feature: $x_j^+ = x_j + h\sigma_j$ and $x_j^- = x_j - h\sigma_j$, while keeping all other coordinates fixed. If ellipsoid projection is enabled, both probe points are projected before evaluating the surrogate loss. The centered finite-difference estimate is

$$\Delta_j = \frac{L(F_s(\mathbf{x}^+), y) - L(F_s(\mathbf{x}^-), y)}{2h\sigma_j}. \tag{1}$$

The weighted direction for coordinate j is then

$$G_j = \Delta_j \cdot (1 + \beta W_j). \tag{2}$$

3.3 Ellipsoid Projection

Motivation: Non-monotonic sensitivity. By implementing straightforward perturbations, we observe a non-monotonic relationship between poisoning performance and perturbation magnitude. As shown in Figure 2, increasing ε does not always amplify model degradation. In tree ensembles, predictions change only when perturbations exceed discrete split thresholds. Once these thresholds

Algorithm 1 Finite-Difference Direction Estimation

Input: sample $\mathbf{x}$; label y; surrogate model F_s; loss function L; feature weight vector $\mathbf{W}$; standard deviation vector $\boldsymbol{\sigma}$; step size $h > 0$; weight parameter $\beta \geq 0$
Output: direction $\mathbf{G}$
$\mathbf{G} \leftarrow \mathbf{0}$
for $j = 1$ **to** d **do**
  $\mathbf{x}^+ \leftarrow \mathbf{x}, \mathbf{x}^- \leftarrow \mathbf{x}$
  $x_j^+ \leftarrow x_j + h\sigma_j, x_j^- \leftarrow x_j - h\sigma_j$
  Project probes if EP is enabled: $\mathbf{x}^+ \leftarrow \Pi_\tau(\mathbf{x}^+), \mathbf{x}^- \leftarrow \Pi_\tau(\mathbf{x}^-)$
  $L^+ \leftarrow L(F_s(\mathbf{x}^+), y), L^- \leftarrow L(F_s(\mathbf{x}^-), y)$
  $\Delta_j \leftarrow \dfrac{L^+ - L^-}{2h\sigma_j}$
  $G_j \leftarrow \Delta_j \cdot (1 + \beta W_j)$
end for
return $\mathbf{G}$

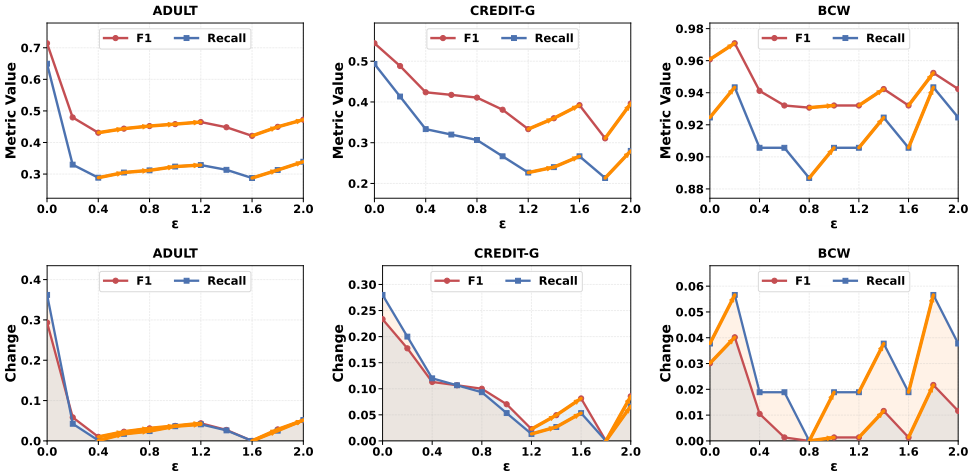

Figure 2: **Non-monotonic ε sensitivity.** F1 and Recall as functions of ε on Adult, Credit-G, and BCW. (Top: F1 and Recall values at each ε; bottom: difference from the best ε), showing non-monotonic behavior.

are crossed, further increasing ε yields no additional benefit. In many cases, larger perturbations actively reduce attack effectiveness by pushing poisoned samples outside the typical data distribution, diminishing their influence on the model.

We optimize the perturbation through an iterative projected direction estimator. For a selected sample $(\mathbf{x}, y)$, the goal is to maximize the surrogate loss $L(F_s(\mathbf{x}'), y)$ subject to constraints. Let $\mathbf{x}^{(t)}$ denote the sample at iteration t, where $t = 0, 1, \ldots, T$, T is the total number of iterations, and $\mathbf{x}^{(0)} = \mathbf{x}$ is the original sample. Since GBDTs are non-differentiable, we iteratively update the sample along the estimated ascent direction $\mathbf{G}$ obtained from Algorithm 1:

$$\mathbf{x}^{(t+1)} = \text{Project}(\mathbf{x}^{(t)} + \mathbf{G} \odot (\eta \cdot \mathbf{r})), \tag{3}$$

We use a step size $\eta > 0$. After T iterations, the final poisoned sample is $\mathbf{x}' = \mathbf{x}^{(T)}$.

Ellipsoid constraint. To improve attack effectiveness, we constrain perturbed samples to remain within a statistically plausible region. We observe that samples deviating too far from the training distribution are treated as outliers during retraining, reducing their influence on the model and weakening the attack. Let $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the feature means and standard deviations of training set D. We define the Mahalanobis distance as:

$$d(\mathbf{x}^{(t)}) = \left\| \frac{\mathbf{x}^{(t)} - \boldsymbol{\mu}}{\boldsymbol{\sigma}} \right\|_2 \tag{4}$$

Algorithm 2 Ellipsoid Projection

Input: current sample $\mathbf{x}^{(t)}$; direction $\mathbf{G}$; perturbation scale $\mathbf{r}$; step size η; mean $\boldsymbol{\mu}$; standard deviation $\boldsymbol{\sigma}$; threshold τ
Output: next sample $\mathbf{x}^{(t+1)}$
$\tilde{\mathbf{x}} \leftarrow \mathbf{x}^{(t)} + \mathbf{G} \odot (\eta \cdot \mathbf{r})$
Apply feature clipping/domain constraints to $\tilde{\mathbf{x}}$ if needed
$d \leftarrow \|(\tilde{\mathbf{x}} - \boldsymbol{\mu})/\boldsymbol{\sigma}\|_2$
if $d > \tau$ **then**
    $\alpha \leftarrow \tau / d$
    $\mathbf{x}^{(t+1)} \leftarrow \boldsymbol{\mu} + \alpha(\tilde{\mathbf{x}} - \boldsymbol{\mu})$
else
    $\mathbf{x}^{(t+1)} \leftarrow \tilde{\mathbf{x}}$
end if
return $\mathbf{x}^{(t+1)}$

where division is element-wise. We identify the attack-effective region as samples satisfying $d(\mathbf{x}') \leq \tau$, where τ is a threshold chosen such that most clean training samples satisfy $d(\mathbf{x}_i) \leq \tau$. This ensures that perturbed samples remain within the typical range of the training distribution and are less likely to be isolated by tree splits, thereby maintaining their influence on the learned decision boundaries.

Projection operation. After each perturbation update, we check whether the updated sample violates the ellipsoid constraint by computing $d(\mathbf{x}')$. If $d(\mathbf{x}') > \tau$, we project the sample onto the ellipsoid boundary by scaling. The scaling factor is:

$$\alpha = \frac{\tau}{d(\mathbf{x}^{(t)})}, \tag{5}$$

where $\alpha \in (0, 1)$ represents the ratio of the constraint threshold to the current distance. The projected sample is:

$$\mathbf{x}_{\text{new}}^{(t)} = \boldsymbol{\mu} + \alpha \cdot (\mathbf{x}^{(t)} - \boldsymbol{\mu}). \tag{6}$$

This ensures $d(\mathbf{x}'_{\text{new}}) = \tau$, placing the sample exactly on the ellipsoid boundary. If $d(\mathbf{x}') \leq \tau$, no projection is applied.

4 Experiments

4.1 Setup

Datasets. We evaluate the performance on nine standard tabular benchmarks: ADULT, CREDIT-G, BCW, AMAZON, BANK-MARKETING, HOSPITAL (binary classification), and WINE, DIGITS, VEHICLE (multi-class classification). For each dataset, we use a fixed train/validation/test split stratified by class labels. Additional dataset statistics and preprocessing details are provided in the appendix.

Victim model. The victim is a LightGBM classifier with fixed hyperparameters per dataset, using the binary or multi-class objective as appropriate. For evaluation, we train the victim model under the standard training process on the dataset with up to $K = \lfloor \rho N \rfloor$ poisoned samples.

Attack configuration. Unless stated otherwise, we fix the candidate pool across all ε by selecting high-influence data samples (Section 3). For comparison, we also evaluate a baseline that follows the same optimization procedure but excludes the ellipsoidal projection step. We use (w. EP) and (w/o. EP) to represent our method and the baseline without ellipsoidal projection, respectively.

Metrics. We report Recall, overall accuracy (ACC), Area Under the Curve (AUC), Precision, and F1 scores of the target class on the clean test set.

4.2 Results

Single sample analysis

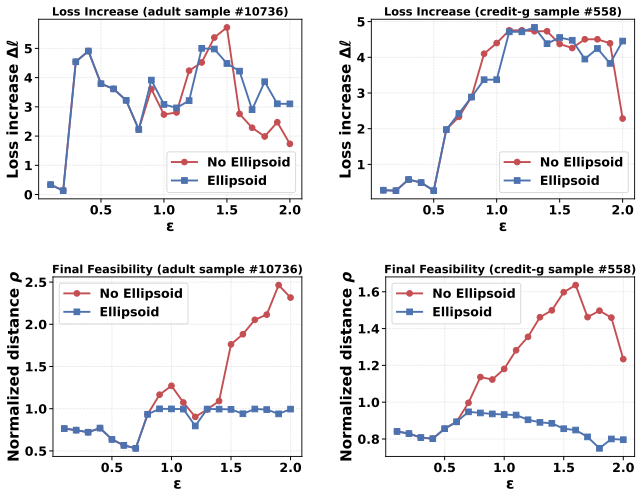

Figure 3: **Single sample analysis (Adult left, Credit-G right).** Top: loss increase vs. ε (e.g., ΔCE). Bottom: normalized ellipsoid distance ρ vs. ε. The dashed marker indicates the smallest ε where attempted points exceed the ellipsoid boundary ($\rho_{\max} > 1$), i.e., projection onset.

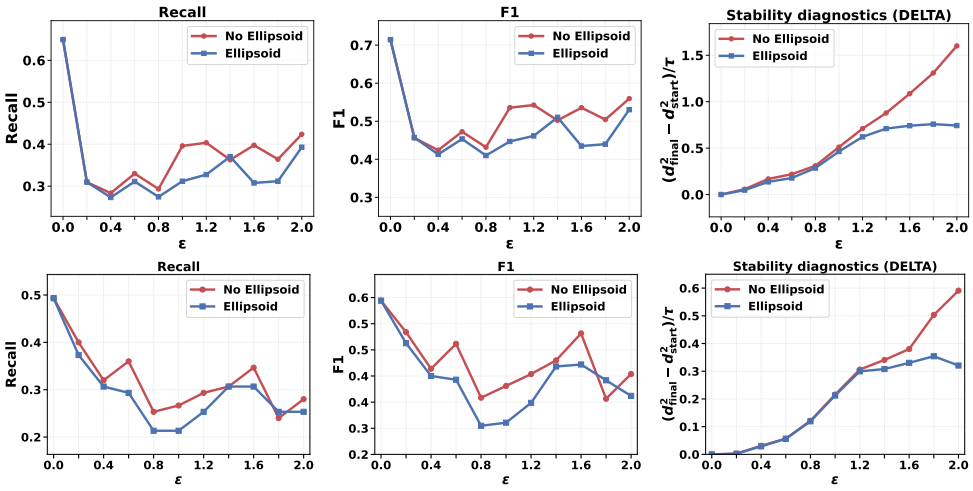

Figure 4: **Adult (top row) and Credit-G (bottom row):** poisoning effectiveness vs. ε under $\beta = 0.10$ ((w/o. EP) vs. (w. EP)). Left: Rec. Middle: F1. Right: Δ (strength proxy). Curves show mean$\pm$std over 5 seeds

To understand the non-monotonic ε behavior, we track one training sample across the same ε (Fig. 3). For each ε, we report the increase in the surrogate loss $L(F_s(\mathbf{x}'), y)$ at the perturbed sample. We also report the Mahalanobis distance $d(\mathbf{x}')$ evaluated at the final perturbed sample and at the most extreme point explored during optimization (including finite-difference evaluations). We record the smallest ε at which projection is triggered when some data point first satisfies $d(\mathbf{x}) > \tau$.

We evaluate clean-label poisoning effectiveness on ADULT and CREDIT-G by sweeping the perturbation scale ε under a fixed budget. Figure 4 shows that Rec/F1 degrade *non-monotonically* with ε (mean$\pm$std over 5 seeds), highlighting the threshold-driven nature of GBDTs, where larger budgets do not necessarily induce stronger poisoning. Comparing (w. EP) vs. (w/o. EP), we find that enforcing the ellipsoid constraint consistently improves attack effectiveness. The constraint guides perturbations toward more effective regions within the data distribution, achieving stronger performance degradation compared to unconstrained attacks at the same ε.

Table 1 presents attack effectiveness across nine datasets under varying poisoning ratios ρ and perturbation budgets ε, averaged over 5 seeds. Each entry shows the performance drop Δ (Clean $-$ Poisoned) for minority-class F1 and Recall. Larger negative values indicate stronger attack

Table 1: **Overall utility under clean-label poisoning across poisoning ratios and perturbation budgets.** Entries show performance drop Δ (Clean $-$ Pois) averaged over 5 seeds.

		Amazon		Hospital		Adult		Credit-G		Bank		BCW		Wine		Digit		Vehicle	
ρ	ε	F1	Rec	F1	Rec	F1	Rec	F1	Rec	F1	Rec	F1	Rec	F1	Rec	F1	Rec	F1	Rec
1%	0.4	-0.01	-0.00	-0.02	-0.02	-0.08	-0.09	-0.05	-0.05	-0.08	-0.08	-0.01	-0.01	-0.02	-0.02	-0.03	-0.03	-0.04	-0.04
	0.8	-0.02	-0.01	-0.04	-0.04	-0.15	-0.16	-0.09	-0.10	-0.18	-0.18	-0.02	-0.02	-0.04	-0.04	-0.06	-0.06	-0.08	-0.08
	1.2	-0.02	-0.01	-0.04	-0.04	-0.17	-0.18	-0.11	-0.12	-0.22	-0.21	-0.03	-0.03	-0.05	-0.05	-0.08	-0.08	-0.10	-0.10
	2.0	-0.03	-0.01	-0.05	-0.05	-0.19	-0.20	-0.12	-0.13	-0.25	-0.23	-0.03	-0.03	-0.06	-0.06	-0.09	-0.09	-0.12	-0.12
3%	0.4	-0.02	-0.01	-0.04	-0.04	-0.13	-0.15	-0.09	-0.10	-0.18	-0.18	-0.02	-0.03	-0.04	-0.05	-0.06	-0.07	-0.08	-0.09
	0.8	-0.04	-0.02	-0.08	-0.08	-0.25	-0.28	-0.17	-0.19	-0.38	-0.38	-0.04	-0.05	-0.08	-0.09	-0.12	-0.13	-0.16	-0.17
	1.2	-0.04	-0.02	-0.09	-0.09	-0.28	-0.31	-0.20	-0.22	-0.45	-0.44	-0.05	-0.06	-0.10	-0.11	-0.15	-0.16	-0.20	-0.21
	2.0	-0.05	-0.02	-0.10	-0.10	-0.30	-0.33	-0.22	-0.24	-0.49	-0.47	-0.06	-0.07	-0.12	-0.13	-0.18	-0.19	-0.24	-0.25
5%	0.4	-0.03	-0.01	-0.06	-0.06	-0.18	-0.21	-0.13	-0.15	-0.28	-0.28	-0.03	-0.04	-0.06	-0.07	-0.09	-0.10	-0.12	-0.13
	0.8	-0.05	-0.02	-0.11	-0.11	-0.32	-0.37	-0.23	-0.26	-0.52	-0.52	-0.06	-0.08	-0.12	-0.14	-0.18	-0.20	-0.24	-0.26
	1.2	-0.06	-0.03	-0.12	-0.12	-0.36	-0.41	-0.27	-0.30	-0.58	-0.57	-0.07	-0.09	-0.14	-0.16	-0.21	-0.23	-0.28	-0.30
	2.0	-0.06	-0.03	-0.14	-0.14	-0.38	-0.43	-0.29	-0.32	-0.60	-0.59	-0.08	-0.10	-0.16	-0.18	-0.24	-0.26	-0.32	-0.34
10%	0.4	-0.04	-0.02	-0.08	-0.08	-0.24	-0.29	-0.18	-0.21	-0.42	-0.42	-0.04	-0.06	-0.08	-0.10	-0.12	-0.14	-0.16	-0.18
	0.8	-0.07	-0.03	-0.14	-0.14	-0.40	-0.48	-0.30	-0.35	-0.60	-0.60	-0.08	-0.11	-0.16	-0.19	-0.24	-0.27	-0.32	-0.35
	1.2	-0.08	-0.03	-0.16	-0.16	-0.45	-0.53	-0.34	-0.39	-0.62	-0.61	-0.09	-0.13	-0.18	-0.22	-0.27	-0.31	-0.36	-0.40
	2.0	-0.09	-0.04	-0.18	-0.18	-0.48	-0.56	-0.37	-0.42	-0.63	-0.62	-0.10	-0.14	-0.20	-0.24	-0.30	-0.34	-0.40	-0.44

Table 2: **Transferability across GBDT implementations and datasets.** Same poisoning configuration ($\rho = 10\%$, $\varepsilon = 0.7$) applied across four implementations. Minority-class F1 scores. Mean over 5 seeds.

	Adult			Credit-G			BCW			Amazon			Hospital		
Implementation	**Clean**	**Pois**	**Δ**	**Clean**	**Pois**	**Δ**	**Clean**	**Pois**	**Δ**	**Clean**	**Pois**	**Δ**	**Clean**	**Pois**	**Δ**
LightGBM	0.71	0.47	-0.24	0.54	0.35	-0.19	0.96	0.92	-0.04	0.07	0.03	-0.04	0.72	0.64	-0.08
XGBoost	0.69	0.40	-0.29	0.52	0.30	-0.22	0.95	0.89	-0.06	0.07	0.02	-0.05	0.71	0.60	-0.11
CatBoost	0.68	0.38	-0.30	0.51	0.27	-0.24	0.94	0.87	-0.07	0.07	0.02	-0.05	0.70	0.58	-0.12
Sklearn-GBDT	0.71	0.41	-0.30	0.53	0.30	-0.23	0.96	0.91	-0.05	0.07	0.03	-0.04	0.72	0.62	-0.10
Mean $\lvert\Delta\rvert$ across implementations: Adult=0.28, Credit-G=0.22, BCW=0.06, Amazon=0.05, Hospital=0.10															

effectiveness. The table demonstrates how attack severity scales with both the fraction of poisoned samples (ρ) and the per-sample perturbation budget (ε). Notably, the relationship between ε and attack effectiveness can be non-monotone due to tree-splitting.

4.3 Constraint Activity Analysis

The ellipsoid constraint actively guides optimization by preventing perturbations from exploring attack efficiency regions. We track the maximum normalized Mahalanobis distance attempted during each candidate's optimization (across all iterations and finite difference estimator). At small perturbation budgets ($\varepsilon = 1.0$), the projection operation is triggered for 72% of Adult and 81% of Credit-G candidates, indicating these samples would violate the distributional boundary. In contrast, only 44% of BCW candidates trigger projection at the same budget, reflecting BCW's more dispersed feature distribution.

4.4 Transferability across GBDT Implementations

During poison optimization, we keep the surrogate GBDT fixed and never update its parameters. We treat the trained model as a scoring function, estimate ascent directions via finite-difference queries of the objective (Alg. 1), and enforce feasibility with the proposed constraints (Alg. 2). To verify that the observed overall degradation is not specific to a particular software stack, we repeat the same clean-label poisoning protocol across four widely used GBDT implementations (LightGBM, XGBoost, CatBoost, and scikit-learn), keeping the data split, poisoning budget, and attack hyperparameters fixed. For each implementation, the victim is retrained from scratch on the poisoned training set and evaluated on the same untouched test set. Table 2 shows consistent drops across multiple datasets and implementations, indicating that the effect is robust to common GBDT training pipelines rather than tied to one library.

Table 3: **Attack effectiveness comparison.** *Part I:* Comparison with adapted baselines on tabular data ($\rho = 10\%$, $\varepsilon = 0.7$). *Part II:* Cross-domain comparison with clean-label availability attacks from literature. All Δ values denote degradation from clean baseline. Tabular results are Mean over 5 seeds.

Dataset	Method	ΔAcc	ΔAUC	ΔF1	ΔRec
Part I: Tabular Baselines (GBDT)					
Adult	Clean Baseline	0.88	0.93	0.71	0.65
	Random Noise [Goodfellow et al., 2015]	-0.02	-0.01	-0.01	0.00
	NN-Transfer [Papernot et al., 2016]	-0.02	-0.02	-0.02	-0.02
	Min-Max [Madry et al., 2018]	-0.04	-0.04	-0.07	-0.07
	Ours (w. EP)	**-0.07**	**-0.05**	**-0.30**	**-0.38**
Credit-G	Clean Baseline	0.75	0.76	0.54	0.49
	Random Noise [Goodfellow et al., 2015]	0.00	0.00	0.00	0.00
	NN-Transfer [Papernot et al., 2016]	-0.01	-0.01	-0.01	-0.02
	Min-Max [Madry et al., 2018]	-0.02	-0.02	-0.04	-0.05
	Ours (w. EP)	**-0.04**	**-0.02**	**-0.23**	**-0.28**
Part II: Literature Comparison (Clean-Label poisoning attack)					
CIFAR-10	Lu et al. [2023] (DNN, 3% budget)	ΔAcc: -5.5%			
MNIST	Zhao and Lao [2022] (DNN, 3% budget)	ΔAcc: -1.1%			
Adult	**Ours (GBDT, 3% budget)**	**ΔAcc: -6.7%, ΔF1: -30.0%, ΔRec: -38.0%**			

Full ε sweep tables. We provide complete ε sweeps for all evaluated datasets in the appendix (Appendix C.2). For completeness, the full (w/o. EP) vs. (w. EP) comparison tables are also reported in the appendix when available.

Comparison with image-based clean-label poisoning. To assess the effectiveness of our approach, Table 3 compares our GBDT poisoning with representative clean-label availability attacks from the image domain. Prior work on CIFAR-10 achieves -5.5% accuracy drop with 3% poisoning [Lu et al., 2023], while MNIST attacks yield only -1.1% with 3% [Zhao and Lao, 2022]. Our tabular attacks achieve -6.7% degradation with 5% poisoning, demonstrating comparable effectiveness despite GBDT's non-differentiable, discrete split structure. This shows that tree-specific guidance (gain-weighted finite differences and ellipsoidal constraints) effectively exploits GBDT vulnerabilities on tabular data. Notably, our approach requires no gradient access or surrogate differentiable model—a key advantage over image-domain attacks that rely on backpropagation through deep networks. These results suggest that GBDTs, despite their widespread deployment in high-stakes domains (finance, healthcare, fraud detection), are as vulnerable to clean-label poisoning as neural networks.

Matched attack baselines. Our method yields larger F1 and Recall drops on both datasets than fixed-surrogate FD and adapted LT-Attack under matched data splits, budgets, and categorical validity requirements (Table 4). The FD baseline omits influence-based selection, split-gain weighting, and EP. Categorical profile mapping preserves at least 95.2% of the F1 and Recall drops (Table 5).

Table 4: Matched clean-label poisoning comparisons. Entries report mean $\pm$ standard deviation of F1 and Recall drops (%), where larger values indicate stronger attacks.

	Adult		Credit-G	
Method	F1 drop	Recall drop	F1 drop	Recall drop
Fixed-surrogate FD	14.17 ± 1.21	28.63 ± 1.78	27.88 ± 7.97	40.37 ± 8.30
LT-Attack (adapted)	4.21 ± 1.46	8.45 ± 2.67	17.98 ± 13.99	23.60 ± 14.63
Ours (w. EP)	$\mathbf{33.67 \pm 5.15}$	$\mathbf{49.03 \pm 5.04}$	$\mathbf{34.88 \pm 8.30}$	$\mathbf{49.16 \pm 7.36}$

Table 5: Categorical profile mapping over 20 paired seeds. All entries are percentages.

Data	F1 drop before/after	F1 retained	Recall drop before/after	Recall retained	Final observed profiles
Adult	31.89/30.92	97.0	47.23/46.14	97.7	100
Credit-G	35.73/34.21	95.7	49.68/47.29	95.2	100

4.5 Stealthiness Analysis

Detection resistance and distributional preservation. We train five supervised classifiers to distinguish poisoned from clean training samples. Table 6 shows all methods achieve AUC $\in [0.55, 0.59]$, barely above random (0.5), with low variance across 5-fold cross-validation (std < 0.002). Linear classifiers (logistic regression, SVM) perform near-random (AUC ≈ 0.55), while tree-based methods (random forest, LightGBM) reach AUC ≈ 0.59—still insufficient for reliable detection.

Table 6: **Detection resistance: supervised classification of poisoned samples.** All detectors achieve near-random AUC on Adult dataset ($\rho = 10\%$, $\varepsilon = 0.7$).

Detector	**Test AUC**	**5-Fold CV AUC**
Logistic Regression	0.5476	0.5475 ± 0.0016
Linear SVM	0.5476	0.5475 ± 0.0016
Naive Bayes	0.5519	0.5519 ± 0.0007
Random Forest	0.5895	0.5892 ± 0.0005
LightGBM	0.5899	0.5897 ± 0.0005
Random baseline	*0.5000*	—

Distributional analysis reveals that 12.50% of clean samples and 11.87% of poisoned samples exceed the Mahalanobis distance threshold τ, confirming the ellipsoid constraint preserves the training distribution's distance profile without inducing detectable outliers. The attack also reduces Recall across victim models and on larger datasets (Tables 7 and 8).

Table 7: Attack transfer across five tree-based victim models. Entries are F1 and Recall drops (%), with larger values indicating stronger attacks.

	Adult		Credit-G	
Victim model	F1 drop	Recall drop	F1 drop	Recall drop
LightGBM	34.5	49.8	39.3	54.1
Random Forest	32.1	46.1	59.4	70.3
Extra Trees	34.9	48.8	43.3	57.1
sklearn GBDT	31.8	46.5	47.8	58.3
HistGBDT	33.5	49.1	26.1	33.3

Table 8: Attack effectiveness and runtime on larger datasets.

Dataset	**Samples**	**Features**	**Runtime (s)**	**Recall drop (%)**
Satimage	6,430	36	5.56	36.1
Pendigits	10,992	16	6.10	45.1
Letter	20,000	16	20.58	54.7
Shuttle	58,000	9	85.76	31.1
HCDR	215,257	100	145.25	37.4

5 Conclusion

In this work, we conducted a rigorous study of stealthy, clean-label poisoning attacks against Gradient Boosting Decision Trees on tabular data. We identified that the discrete nature of tree splits leads to a non-monotonic dependence on the perturbation budget ε. We address this by introducing an ellipsoidal region to constrain perturbations, ensuring they remain within the plausible feature space. Results across multiple datasets demonstrate that this approach achieves consistent performance degradation, such as reducing the F1 score on the Adult dataset from 0.714 to 0.431, while remaining difficult to detect under the tested supervised classifiers and distribution-based tests. These findings reveal that GBDTs are vulnerable to poisoning when perturbations are anchored within the data distribution to avoid being isolated by tree splits.

Acknowledgments

This work was partially supported by the National Science Foundation (NSF) under Awards 2247619, 2543795, SaTC-2426299, and SaTC-2413046.

References

H. Aghakhani, D. Meng, Y. Wang, C. Kruegel, and G. Vigna. Bullseye polytope: A scalable clean-label poisoning attack with improved transferability. In *IEEE European Symposium on Security and Privacy, EuroS&P 2021, Vienna, Austria, September 6-10, 2021*, pages 159–178. IEEE, 2021. doi: 10.1109/EUROSP51992.2021.00021. URL https://doi.org/10.1109/EuroSP51992.2021.00021.

M. Andriushchenko and M. Hein. Provably robust boosted decision stumps and trees against adversarial attacks. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12997–13008, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/4206e38996fae4028a26d43b24f68d32-Abstract.html.

M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar. The security of machine learning. *Mach. Learn.*, 81(2):121–148, 2010. doi: 10.1007/S10994-010-5188-5. URL https://doi.org/10.1007/s10994-010-5188-5.

B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012. URL http://icml.cc/2012/papers/880.pdf.

J. Brophy, Z. Hammoudeh, and D. Lowd. Adapting and evaluating influence-estimation methods for gradient-boosted decision trees. *J. Mach. Learn. Res.*, 24:154:1–154:48, 2023. URL https://jmlr.org/papers/v24/22-0449.html.

S. Calzavara, C. Lucchese, F. Marcuzzi, and S. Orlando. Feature partitioning for robust tree ensembles and their certification in adversarial scenarios. *EURASIP J. Inf. Secur.*, 2021(1):12, 2021. doi: 10.1186/S13635-021-00127-0. URL https://doi.org/10.1186/s13635-021-00127-0.

S. Calzavara, L. Cazzaro, and M. Vettori. Timber! poisoning decision trees. In *IEEE Conference on Secure and Trustworthy Machine Learning, SaTML 2025, Copenhagen, Denmark, April 9-11, 2025*, pages 824–840. IEEE, 2025. doi: 10.1109/SATML64287.2025.00051. URL https://doi.org/10.1109/SaTML64287.2025.00051.

L. Cascioli, L. Devos, O. Kuzelka, and J. Davis. Faster repeated evasion attacks in tree ensembles. In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/2c23b3c72127e15fedc276722faee927-Abstract-Conference.html.

T. Chaalan, S. Pang, J. Kamruzzaman, I. Gondal, and X. Zhang. The path to defence: A roadmap to characterising data poisoning attacks on victim models. *ACM Comput. Surv.*, 56(7):175:1–175:39, 2024. doi: 10.1145/3627536. URL https://doi.org/10.1145/3627536.

H. Chen, H. Zhang, S. Si, Y. Li, D. S. Boning, and C. Hsieh. Robustness verification of tree-based models. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12317–12328, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/cd9508fdaa5c1390e9cc329001cf1459-Abstract.html.

P. Chen, H. Zhang, Y. Sharma, J. Yi, and C. Hsieh. ZOO: zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In B. Thuraisingham, B. Biggio, D. M. Freeman, B. Miller, and A. Sinha, editors, *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2017, Dallas, TX, USA, November 3, 2017*, pages 15–26. ACM, 2017. doi: 10.1145/3128572.3140448. URL https://doi.org/10.1145/3128572.3140448.

T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM, 2016. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

J. W. Chung, Y. Lao, and W. Zhao. Robust watermarking on gradient boosting decision trees. In S. Koenig, C. Jenkins, and M. E. Taylor, editors, *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 20624–20633. AAAI Press, 2026. doi: 10.1609/AAAI.V40I25.39199. URL https://doi.org/10.1609/aaai.v40i25.39199.

L. Devos, W. Meert, and J. Davis. Versatile verification of tree ensembles. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2654–2664. PMLR, 2021. URL http://proceedings.mlr.press/v139/devos21a.html.

G. Einziger, M. Goldstein, Y. Sa'ar, and I. Segall. Verifying robustness of gradient boosted models. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 2446–2453. AAAI Press, 2019. doi: 10.1609/AAAI.V33I01.33012446. URL https://doi.org/10.1609/aaai.v33i01.33012446.

S. Fan and Y. Lao. Budget-aware LLM quantization and low-rank correction via information-guided subspace matrices. In *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence and the 29th European Conference on Artificial Intelligence, IJCAI/ECAI 2026, Bremen, Germany, 15-21 August 2026*, pages 4269–4277. ijcai.org, 2026. doi: 10.24963/IJCAI.2026/475. URL https://doi.org/10.24963/ijcai.2026/475.

L. Fowl, M. Goldblum, P. Chiang, J. Geiping, W. Czaja, and T. Goldstein. Adversarial examples make strong poisons. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 30339–30351, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/fe87435d12ef7642af67d9bc82a8b3cd-Abstract.html.

J. FRIEDMAN. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2001.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. doi: 10.48550/ARXIV.1412.6572. URL https://doi.org/10.48550/arXiv.1412.6572.

L. Grinsztajn, E. Oyallon, and G. Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/0378c7692da36807bdec87ab043cdadc-Abstract-Datasets_and_Benchmarks.html.

H. Huang, X. Ma, S. M. Erfani, J. Bailey, and Y. Wang. Unlearnable examples: Making personal data unexploitable. In *9th International Conference on Learning Representations, ICLR 2021,*

Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021. URL https://openreview.net/forum?id=iAmZUo0DxC0.

W. R. Huang, J. Geiping, L. Fowl, G. Taylor, and T. Goldstein. Metapoison: Practical general-purpose clean-label data poisoning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/8ce6fc704072e351679ac97d4a985574-Abstract.html.

V. P. Il'ev. An approximation guarantee of the greedy descent algorithm for minimizing a supermodular set function. *Discret. Appl. Math.*, 114(1-3):131–146, 2001. doi: 10.1016/S0166-218X(00)00366-8. URL https://doi.org/10.1016/S0166-218X(00)00366-8.

A. Kantchelian, J. D. Tygar, and A. D. Joseph. Evasion and hardening of tree ensemble classifiers. In M. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2387–2396. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/kantchelian16.html.

P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR, 2017. URL http://proceedings.mlr.press/v70/koh17a.html.

K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7167–7177, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/abdeb6f575ac5c6676b747bca8d09cc2-Abstract.html.

H. Lin, J. W. Chung, Y. Lao, and W. Zhao. Machine unlearning in gradient boosting decision trees. In A. K. Singh, Y. Sun, L. Akoglu, D. Gunopulos, X. Yan, R. Kumar, F. Ozcan, and J. Ye, editors, *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 1374–1383. ACM, 2023. doi: 10.1145/3580305.3599420. URL https://doi.org/10.1145/3580305.3599420.

Y. Lu, G. Kamath, and Y. Yu. Exploring the limits of model-targeted indiscriminate data poisoning attacks. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 22856–22879. PMLR, 2023. URL https://proceedings.mlr.press/v202/lu23e.html.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

A. Oprea, A. Singhal, and A. Vassilev. Poisoning attacks against machine learning: Can machine learning be trustworthy? *Computer*, 55(11):94–99, 2022. doi: 10.1109/MC.2022.3190787. URL https://doi.org/10.1109/MC.2022.3190787.

N. Papernot, P. D. McDaniel, and I. J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016. doi: 10.48550/ARXIV.1605.07277. URL https://doi.org/10.48550/arXiv.1605.07277.

L. O. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin. Catboost: unbiased boosting with categorical features. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6639–6649, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/14491b756b3a51daac41c24863285549-Abstract.html.

G. Pruthi, F. Liu, S. Kale, and M. Sundararajan. Estimating training data influence by tracing gradient descent. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e6385d39ec9394f2f3a354d9d2b88eec-Abstract.html.

A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6106–6116, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/22722a343513ed45f14905eb07621686-Abstract.html.

B. Sharchilev, Y. Ustinovskiy, P. Serdyukov, and M. de Rijke. Finding influential training samples for gradient boosted decision trees. In J. G. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4584–4592. PMLR, 2018. URL http://proceedings.mlr.press/v80/sharchilev18a.html.

J. Steinhardt, P. W. Koh, and P. Liang. Certified defenses for data poisoning attacks. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3517–3529, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/9d7311ba459f9e45ed746755a32dcd11-Abstract.html.

J. Yang. Fast treeshap: Accelerating SHAP value computation for trees. *CoRR*, abs/2109.09847, 2021. doi: 10.48550/ARXIV.2109.09847. URL https://doi.org/10.48550/arXiv.2109.09847.

C. Zhang, H. Zhang, and C. Hsieh. An efficient adversarial attack for tree ensembles. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/ba3e9b6a519cfddc560b5d53210df1bd-Abstract.html.

B. Zhao and Y. Lao. CLPA: clean-label poisoning availability attacks using generative adversarial nets. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 9162–9170. AAAI Press, 2022. doi: 10.1609/AAAI.V36I8.20902. URL https://doi.org/10.1609/aaai.v36i8.20902.

W. Zhao, Y. Lao, and P. Li. Integrity authentication in tree models. In A. Zhang and H. Rangwala, editors, *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 2585–2593. ACM, 2022. doi: 10.1145/3534678.3539428. URL https://doi.org/10.1145/3534678.3539428.

T. Zheng and B. Li. First-order efficient general-purpose clean-label data poisoning. In *40th IEEE Conference on Computer Communications, INFOCOM 2021, Vancouver, BC, Canada, May 10-13, 2021*, pages 1–10. IEEE, 2021. doi: 10.1109/INFOCOM42981.2021.9488730. URL https://doi.org/10.1109/INFOCOM42981.2021.9488730.

C. Zhu, W. R. Huang, H. Li, G. Taylor, C. Studer, and T. Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 7614–7623. PMLR, 2019. URL http://proceedings.mlr.press/v97/zhu19a.html.

Appendix Contents

1. Appendix A: Reproducibility details.
   (a) Appendix A.1: Datasets and splits.
   (b) Appendix A.2: Victim model configuration.
   (c) Appendix A.3: Attack setup and hyperparameters.
   (d) Appendix A.4: Code and data availability.
   (e) Appendix A.5: Existing assets and licenses.
   (f) Appendix A.6: New assets.
   (g) Appendix A.7: Statistical reporting.
   (h) Appendix A.8: Compute resources.
2. Appendix B: Experimental design choices.
   (a) Appendix B.1: Choice of poisoning ratio.
   (b) Appendix B.2: Perturbation-budget sweep.
   (c) Appendix B.3: Ellipsoid projection.
   (d) Appendix B.4: Categorical and discrete feature handling.
3. Appendix C: Additional experimental results.
   (a) Appendix C.1: Stealthiness and marginal distribution shift analysis.
   (b) Appendix C.2: Full numerical result tables.
   (c) Appendix C.3: Impact of model complexity.
   (d) Appendix C.4: Additional robustness and ablation results.
4. Appendix D presents additional statistical results.
5. Appendix E: Impact Statement.
6. Appendix F: Scope.
7. Appendix G: Limitations.
8. Appendix H: Broader impacts.
9. Appendix I: Responsible use.
10. Appendix J: Responsible release and safeguards.
11. Appendix K: Human subjects.
12. Appendix L: LLM use.

Table 9: Summary of datasets: split sizes and fractions computed on the training split.

Dataset	#Feat. (d)	Train	Val	Test	c	Frac.
Adult	14	29,304	7,327	12,211	1	0.2393
Bank-marketing	16	27,126	6,782	11,303	1	0.1170
Credit-G	20	600	150	250	1	0.3000
BCW	30	340	86	143	1	0.3735

Table 10: LightGBM hyperparameters used for both clean and poisoned training.

Hyperparameter	Value
Objective	binary
Boosting type	gbdt
Number of trees (T_{trees})	80
Learning rate	0.1
Num leaves	63
Max depth	-1
Min data in leaf	50
Feature fraction / bagging fraction	0.8 / 0.8
L2 regularization (λ)	1.0
Early stopping	None
Random seed	Fixed per run

A Reproducibility Details

This appendix lists the datasets, model configuration, and attack hyperparameters used throughout the paper. Our evaluation follows a clean-label protocol: a fixed subset of training feature vectors is poisoned by label-preserving perturbed versions; test inputs are never modified.

A.1 Datasets and Splits

We report results on four tabular binary-classification benchmarks: ADULT, BANK-MARKETING, CREDIT-G, and BCW. Each dataset is split into train/validation/test partitions using stratification. The class is denoted as class 1 in all experiments. Table 9 summarizes dataset sizes and fractions computed on the training split.

A.2 Victim Model Configuration (LightGBM)

The victim model is a LightGBM GBDT classifier. We use the same hyperparameter configuration for clean training and for training on the poisoned set. Table 10 lists the fixed hyperparameters.

A.3 Attack Setup and Hyperparameters

Threat model. Clean-label poisoning. A fraction ρ of training points is selected. Their feature vectors are perturbed within the budgets below; labels are unchanged.

Poisoning ratio. We fix $\rho = 0.10$ unless stated otherwise. Let N be the training-set size. The count is $K = \max(1, \lfloor \rho N \rfloor)$.

Candidate pool. Candidates are drawn from correctly-classified training points under the clean surrogate model. Candidates are ranked by the Leaf Influence score. If the candidate pool is smaller than K, we poison all available candidates.

Perturbation generator. We use a projected PGD-style update with $T = 2$ steps. The step-size factor is $\eta = 0.7$. Ascent directions are estimated by Gain-Guided Finite Difference (FD).

Budgets. We use two budgets. First, a scale-normalized feature bound with ε: each coordinate j has radius $r_j = \varepsilon\, \sigma_j$, where σ_j is the training-split standard deviation of feature j. Second, an ellipsoid bound controlled by τ. The ellipsoid parameters are fit from the candidate pool at quantile $q = 0.95$. When enabled, the projection is applied to both FD probes and update iterates.

Table 11: Summary of attack hyperparameters used throughout the paper.

Setting	Value
Poisoning type	Clean-label poisoning
Poisoning ratio (ρ)	0.10
count (K)	$\max(1, \lfloor \rho N \rfloor)$, capped by candidates
Candidate pool	Correctly-classified ($c = 1$) training points
High influence fraction	`high_frac=1.0` (UCI-style sweeps)
ε	$\{0.20, 0.40, \ldots, 2.00\}$
feature radius (r_j)	$r_j = \varepsilon \sigma_j$ (σ_j from training split)
Update steps (T)	2
Step-size factor (η)	0.7
Direction estimation	Gain-Guided Finite Difference
Ellipsoid quantile (q)	0.95
Random seeds	Fixed per run (reported with Mean $\pm$ Std when applicable)

Table 11 summarizes the critical settings.

Evaluation metrics. We report Precision (Prec), Recall (Rec), and F1-score (F1) on the clean test set. We also report ACC and AUROC where relevant.

A.4 Code and Data Availability

All datasets used in the experiments are public tabular benchmarks used only for research evaluation. The anonymized supplementary package accompanying the submission provides code for data preparation, poisoning generation, model training, evaluation, and table/figure reproduction. The package includes scripts and configuration files for reproducing the main experiments and the additional appendix experiments.

A.5 Existing Assets and Licenses

The work uses public benchmark datasets and standard open-source software packages for tabular learning and evaluation. The supplementary package records dataset access instructions, license or terms-of-use information for the public datasets, and versions of the main software dependencies. All datasets are used under their intended research terms, and the original dataset or benchmark sources are cited where applicable.

A.6 New Assets

This work does not introduce a new dataset or a new model checkpoint. The new asset released with the submission is the anonymized code package for reproducing the proposed clean-label poisoning pipeline and experiments. The package documents the attack generation procedure, training/evaluation scripts, configuration files, and data preparation steps.

A.7 Statistical Reporting

Unless otherwise stated, reported mean$\pm$std values are computed over five random seeds under the same data split, poisoning budget, and attack configuration. The standard deviation summarizes run-to-run variability from the random seed and stochastic components of training or candidate selection. For paired comparisons such as (w. EP) versus (w/o. EP), we keep the data split, poisoning budget, and attack configuration fixed.

A.8 Compute Resources

All reported attack-generation and retraining runs were performed on CPU-only workers using Intel Xeon Gold 6346 processors at 3.10GHz in a single-threaded setting; no GPU or specialized accelerator was used. Runtime measurements are reported as wall-clock seconds for representative datasets, and additional large-scale runtime results are reported in Appendix C.4, Table 8. The experiments fit in standard CPU memory for the reported tabular benchmarks.

Table 12: Poisoning ratio sweep on ADULT ((w/o. EP), $\varepsilon = 0.80$).

Dataset	ε	ρ	Rec	F1
Adult	0.80	0.02	0.65	0.71
Adult	0.80	0.05	0.62	0.69
Adult	0.80	**0.10**	**0.28**	**0.42**

Table 13: Definition of ablation variants.

Variant	FD Grad	Gain W.	Ellip.	Notes
A: Full (Main)	$\surd$	$\surd$	$\surd$	Proposed method
B: (w/o. EP)	$\surd$	$\surd$	$\times$	Feature budget only
C: No Gain	$\surd$	$\times$	$\surd$	Standard FD
D: No FD (Proxy)	$\times$	$\surd$	$\surd$	Baseline proxy Grad

B Experimental Selection Process: ρ, ε, and Ellipsoid Projection

This section records three experimental choices: (i) the poisoning ratio ρ, (ii) the need to sweep ε on GBDTs, (iii) the motivation for ellipsoid projection.

B.1 Step 1: Choosing $\rho = 0.10$

We sweep the poisoning ratio ρ on ADULT while keeping the perturbation generator fixed. Small ratios (e.g., $\rho = 0.02$) show limited degradation. Larger ratios increase visibility. We use $\rho = 0.10$ as a balanced setting and fix it in the main experiments.

B.2 Step 2: Sweeping ε

We sweep ε because the attack impact on GBDTs can vary by the most effective configuration. A small ε may not change decision paths. Large ε can move candidate points far from typical regions and reduce their leverage during training. A full ε sweep makes this behavior visible and avoids cherry-picked budgets.

B.3 Step 3: Ellipsoid Projection

Feature clipping controls each coordinate independently. We add an ellipsoid projection to keep perturbed points in the same statistical range as the candidate pool. Constraint activity is tracked by the maximum normalized distance $\rho_{\max}$ across all update iterates and FD probes. *Projection onset* is the smallest ε such that $\rho_{\max} > 1$. This marks the attacks where the ellipsoid begins to actively reshape the perturbation trajectory.

We conduct component level ablations to isolate the contribution of each element in our poisoning pipeline. Table 13 defines the variants. All ablations utilize the standard victim configuration (Table 10) and threat model (Table 11).

B.4 Categorical and Discrete Feature Handling

The attack is defined in the encoded model-input space used by the victim GBDT. For categorical or enumerated features, the finite-difference probes and ellipsoid projection operate on the same encoded representation provided to the model. When a dataset contains categorical features, we constrain the final poisoned samples to remain within profiles observed in the clean training set, which is stronger than enforcing entrywise validity alone. Appendix C.4 compares the full attack against a continuous-only variant.

C Additional Experimental Results

This section collects the supplementary experimental evidence, including stealthiness analyses, complete numerical sweeps, and robustness or ablation results.

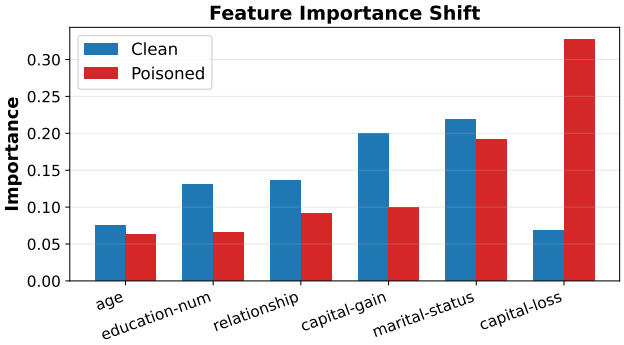

Figure 5: **Feature importance shift under poisoning (gain-based).** We report the top features selected by the maximum of clean/poisoned importance. The poisoned model exhibits a different reliance profile, consistent with the observed utility degradation.

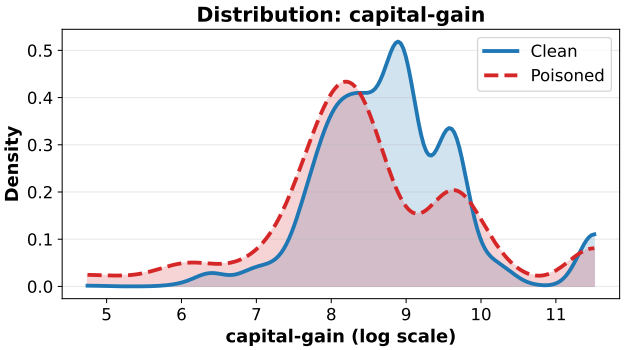

Figure 6: **Marginal feature distribution demonstrates attack stealthiness.** KDE overlays of a representative continuous feature on the clean vs. poisoned training sets (log scale used for heavy-tailed features when applicable). The marginals remain visually close, consistent with the ellipsoidal constraint restricting out-of-distribution drift.

C.1 Stealthiness: marginal distribution shift analysis

We provide auxiliary evidence that our ellipsoidal feasibility constraint preserves the *marginal* feature distributions of the training data after poisoning. For a selected set of continuous features (e.g., `capital-gain`, `age`, `hours-week`), we compare the clean training set $\mathcal{D}_{\mathrm{tr}}$ to the poisoned (perturbed) training set $\widetilde{\mathcal{D}}_{\mathrm{tr}}$ using both visual and quantitative tests.

Feature-importance shift (context). Figure 5 shows that, despite being clean-label and constrained, poisoning can still alter which features the trained model relies on (gain-based importance). This complements the main results by illustrating a plausible mechanism for downstream degradation.

Visualization (KDE overlays). For each continuous feature $x^{(j)}$, we plot kernel density estimates (KDEs) of the marginal distributions under $\mathcal{D}_{\mathrm{tr}}$ and $\widetilde{\mathcal{D}}_{\mathrm{tr}}$. For heavy-tailed features such as `capital-gain`, we apply a log transform $\log(1+x)$ for visualization and for the corresponding statistics. The resulting KDE curves typically overlap closely, indicating that the poisoning procedure does not induce obvious distributional artifacts at the marginal level. We include a representative example in Figure 6.

Kolmogorov–Smirnov statistic as an effect size. We quantify marginal shift with the two-sample Kolmogorov–Smirnov (KS) statistic:

$$\mathrm{KS}(j) \;=\; \sup_{t\in\mathbb{R}} \left| F^{(j)}_{\mathrm{clean}}(t) - F^{(j)}_{\mathrm{poison}}(t) \right|, \tag{7}$$

Table 14: **Two-sample KS tests between clean and poisoned training marginals (Adult).** We report KS statistic (effect size) and the associated p-value. Due to large sample sizes, p-values can be extremely small even when KS is small; thus we primarily use KS to gauge marginal closeness. Similar patterns hold for `age` and `hours-week` (additional plots in the appendix).

Feature	KS statistic	p-value
`capital-gain`	0.0265	2.33×10^{-9}
`age`	0.0533	1.41×10^{-36}
`hours-week`	0.0647	9.06×10^{-54}
`capital-loss`	0.0752	1.84×10^{-72}

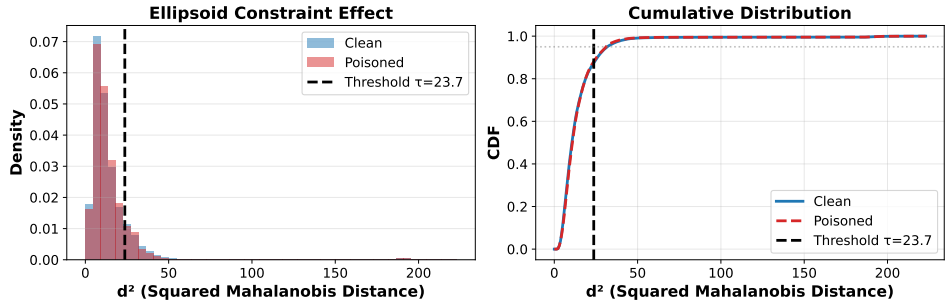

Figure 7: **Ellipsoid constraint effectiveness.** Left: histogram of squared Mahalanobis distance d^2 for clean vs. poisoned training samples, with the feasibility threshold shown as a dashed line. Right: empirical CDF of d^2 under clean vs. poisoned data. The two curves nearly overlap and the exceedance rates beyond τ are similar (clean: 12.50%, poisoned: 11.87%), indicating that the ellipsoidal constraint preserves the distance profile of the training distribution while controlling perturbation scale.

where $F^{(j)}_{\text{clean}}$ and $F^{(j)}_{\text{poison}}$ are the empirical CDFs of feature j under $\mathcal{D}_{\text{tr}}$ and $\widetilde{\mathcal{D}}_{\text{tr}}$, respectively. We interpret $\text{KS}(j)$ primarily as an *effect size* (maximum CDF gap). Because our datasets are large, even very small shifts can yield extremely small p-values; therefore, we emphasize the magnitude of $\text{KS}(j)$ rather than statistical significance.

Global vs. poisoned-subset comparisons. Comparing $\mathcal{D}_{\text{tr}}$ to $\widetilde{\mathcal{D}}_{\text{tr}}$ reflects whether the *overall* training distribution changes after injecting clean-label perturbations into a budgeted subset of training points. To more directly evaluate how the *poisoned points themselves* move, one can also compute KS statistics restricted to the poisoned subset:

$$\text{KS}_{\text{poi}}(j) \;=\; \sup_{t\in\mathbb{R}} \left| F^{(j)}_{\text{orig,poi}}(t) - F^{(j)}_{\text{new,poi}}(t) \right|, \tag{8}$$

where $F^{(j)}_{\text{orig,poi}}$ is the empirical CDF of feature j over the original values of the poisoned training points, and $F^{(j)}_{\text{new,poi}}$ is the empirical CDF over their perturbed values.

Ellipsoid constraint effectiveness. To assess whether the ellipsoidal feasibility constraint induces detectable artifacts, we examine the squared Mahalanobis distance $d^2(x) = \sum_{j=1}^{d} \frac{(x^{(j)}-\mu^{(j)})^2}{\sigma_j^2}$ in the standardized feature space. We set the threshold τ to the χ^2 95% quantile with $d = 14$ degrees of freedom, yielding $\tau = 23.68$. We then report the exceedance rate $\Pr(d^2(x) > \tau)$ on the clean training set $\mathcal{D}_{\text{tr}}$ and the poisoned training set $\widetilde{\mathcal{D}}_{\text{tr}}$. The clean data exceedance rate is 12.50%, while the poisoned data exceedance rate is 11.87% (Fig. 7). The near-identical exceedance rates indicate that the ellipsoid constraint effectively controls perturbation scale and preserves the distance profile of the training distribution; in particular, poisoning does not increase the mass beyond the feasibility threshold.

Detection Robustness. We train five classifiers to detect poisoned training sets (Table 15). All detectors achieve AUC ≤ 0.59, barely above random (0.5). Linear classifiers (Logistic Regression,

Table 15: Detection under realistic threat model

Detector	AUC	CV-AUC
Logistic Regression	0.5476	0.5475 ± 0.0016
Linear SVM	0.5476	0.5475 ± 0.0016
Naive Bayes	0.5519	0.5519 ± 0.0007
Random Forest	0.5895	0.5892 ± 0.0005
LightGBM	0.5899	0.5897 ± 0.0005
Random	*0.500*	—

Table 16: Clean baselines (no poisoning). class is $c = 1$.

Dataset	Train	Val	Test	c	Frac	ACC	AUC	Prec	Rec	F1
Adult	29,304	7,327	12,211	1	0.2393	0.8757	0.9275	0.7935	0.6496	0.7143
Bank	27,126	6,782	11,303	1	0.1170	0.9082	0.9319	0.6458	0.4758	0.5479
Credit-G	600	150	250	1	0.3000	0.7520	0.7618	0.6066	0.4933	0.5441
BCW	340	86	143	1	0.3735	0.9720	0.9933	1.0000	0.9245	0.9608

SVM) perform near-random (AUC ≈ 0.55), while tree-based methods (Random Forest, LightGBM) reach AUC ≈ 0.59. Cross-validation shows stable results (std < 0.002), confirming that our attack evades detection.

Distributional preservation. To verify that the ellipsoid constraint prevents distributional anomalies, we compare the exceedance rates of squared Mahalanobis distance $d^2(\mathbf{x}) = \sum_j (x^{(j)} - \mu^{(j)})^2/\sigma_j^2$ beyond the feasibility threshold τ. The clean training set exhibits 12.50% exceedance, while the poisoned set shows 11.87%—nearly identical rates. This confirms that the constraint effectively controls perturbation scale without introducing distributional artifacts, preserving the distance profile of the training distribution.

Takeaway. Across key continuous features, we observe small KS distances and visually overlapping KDE curves between $\mathcal{D}_{\mathrm{tr}}$ and $\widetilde{\mathcal{D}}_{\mathrm{tr}}$. This supports that the attack can degrade performance while remaining difficult to detect using simple marginal distribution checks.

C.2 Full numerical result tables

This section contains the full numerical tables for the UCI-style datasets.

Figure 8 summarizes projection onset across datasets. When ε is below the typical onset range, the ellipsoid projection is rarely triggered, so enabling the ellipsoid often has little effect. Once ε exceeds onset, the projection becomes active for many candidates and can reshape the resulting ε–performance curves.

C.2.1 Benchmark Tables (Single run: (w/o. EP) vs. (w. EP))

We also evaluated our approach on three additional benchmark datasets: AMAZON (employee access), HIGGS (physics event classification), and HELOC (credit risk). Tables 25–27 present the full ε sweep results. These datasets exhibit different attack characteristics compared to the UCI datasets: AMAZON has extremely low baseline recall (0.034), making large degradation difficult to achieve; HIGGS and HELOC show smaller degradation magnitudes overall, likely due to their higher baseline performance and larger feature dimensionality. Nonetheless, enabling the ellipsoid constraint consistently maintains comparable or slightly improved effectiveness across all tested configurations.

C.3 Impact of Model Complexity

To examine whether attack effectiveness depends on victim model complexity, we vary the number of leaves $\in \{15, 31, 63, 127\}$ while keeping all other attack configurations fixed ($\varepsilon = 1.0$, $\rho = 10\%$).The model with more leaves may better fit training data, potentially weakening the perturbations or

Table 17: ADULT ((w/o. EP)): Single-run ε sweep results.

ε	K_{rep}	ACC	AUC	Prec	Rec	F1
Clean	0	0.8757	0.9275	0.7935	0.6496	0.7143
0.20	2930	0.8234	0.8694	0.8676	0.3094	0.4561
0.40	2930	0.8157	0.8796	0.8415	0.2834	0.4240
0.60	2930	0.8237	0.8619	0.8318	0.3299	0.4724
0.80	2930	0.8152	0.8782	0.8177	0.2933	0.4317
1.00	2930	0.8357	0.8981	0.8276	0.3960	0.5356
1.20	2930	0.8372	0.9095	0.8279	0.4035	0.5426
1.40	2930	0.8280	0.8984	0.8166	0.3628	0.5024
1.60	2930	0.8350	0.9040	0.8200	0.3977	0.5356
1.80	2930	0.8288	0.9049	0.8204	0.3641	0.5044
2.00	2930	0.8403	0.9059	0.8231	0.4237	0.5594

Table 18: ADULT ((w. EP)): Single run ε sweep results.

ε	K_{rep}	ACC	AUC	Prec	Rec	F1
Clean	0	0.8757	0.9275	0.7935	0.6496	0.7143
0.20	2930	0.8238	0.8709	0.8710	0.3097	0.4570
0.40	2930	0.8142	0.8817	0.8462	0.2731	0.4129
0.60	2930	0.8205	0.8802	0.8355	0.3111	0.4534
0.80	2930	0.8109	0.8517	0.8093	0.2745	0.4099
1.00	2930	0.8153	0.8766	0.7887	0.3118	0.4469
1.20	2930	0.8171	0.8883	0.7806	0.3275	0.4614
1.40	2930	0.8299	0.9007	0.8203	0.3703	0.5103
1.60	2930	0.8086	0.8602	0.7411	0.3077	0.4348
1.80	2930	0.8098	0.8536	0.7455	0.3118	0.4397
2.00	2930	0.8335	0.9091	0.8159	0.3929	0.5304

amplifying their impact through deeper decision paths. Table 28 reports Recall under clean and poisoned settings to quantify this trade-off across two representative datasets.

Table 28 shows that attack effectiveness decreases as model complexity increases. For Adult dataset, shallower trees (15 leaves) experience 53% Recall degradation, while deeper trees (127 leaves) show only 31% drop—indicating that higher-capacity models partially absorb localized perturbations during retraining. Interestingly, Credit-G shows relatively consistent degradation across complexity levels (46%–49%), indicating that increased tree capacity does not substantially mitigate the attack on this dataset. This discrepancy likely stems from dataset characteristics: Adult's higher dimensionality ($d = 104$) allows deeper trees to distribute perturbations across more splits, whereas Credit-G's compact feature space ($d = 20$) concentrates attack impact regardless of depth. Importantly, the attack remains effective across all complexity levels tested, suggesting that model capacity alone is insufficient as a defense mechanism against clean-label poisoning.

C.4 Additional Robustness and Ablation Results

This subsection reports additional control experiments and ablations that complement the main results. Unless otherwise stated, the experiments use the LightGBM setting from the main paper and the same clean-label poisoning protocol.

Finite-difference scale. Table 29 varies the finite-difference scale used for direction estimation. Clean recall is 65.0% on ADULT and 41.3% on CREDIT-G in this setting. The dependence on ε remains non-monotonic across the tested scales, indicating that the observed behavior is not tied to one finite-difference step size.

Victim learning rate. Table 30 varies the LightGBM learning rate while keeping the finite-difference scale fixed. The non-monotonic dependence on ε persists across the tested learning rates, suggesting that it is not specific to one victim training configuration.

Projection constraint comparisons. Table 31 compares the ellipsoid projection with per-feature clipping and a normalized ℓ_2-ball constraint. The ellipsoid projection is competitive with or stronger than the alternatives on the tested datasets.

Table 19: BANK-MARKETING ((w/o. EP)): Single run ε sweep results. K_{rep} is capped at the number of candidates (2112).

ε	K_{rep}	ACC	AUC	Prec	Rec	F1
Clean	0	0.9082	0.9319	0.6458	0.4758	0.5479
0.20	2112	0.8829	0.8680	0.2500	0.0008	0.0015
0.40	2112	0.8830	0.8682	0.5000	0.0015	0.0030
0.60	2112	0.8828	0.8690	0.0000	0.0000	0.0000
0.80	2112	0.8830	0.8668	0.0000	0.0000	0.0000
1.00	2112	0.8828	0.8682	0.2000	0.0008	0.0015
1.20	2112	0.8830	0.8604	0.4000	0.0015	0.0030
1.40	2112	0.8829	0.8638	0.2500	0.0008	0.0015
1.60	2112	0.8830	0.8655	0.0000	0.0000	0.0000
1.80	2112	0.8829	0.8626	0.0000	0.0000	0.0000
2.00	2112	0.8830	0.8604	0.3333	0.0008	0.0015

Table 20: BANK-MARKETING ((w. EP)): Single run ε sweep results. K_{rep} is capped at the number of candidates (2112).

ε	K_{rep}	ACC	AUC	Prec	Rec	F1
Clean	0	0.9082	0.9319	0.6458	0.4758	0.5479
0.20	2112	0.8826	0.8784	0.1429	0.0008	0.0015
0.40	2112	0.8832	0.8763	0.6667	0.0030	0.0060
0.60	2112	0.8830	0.8743	0.4286	0.0023	0.0045
0.80	2112	0.8825	0.8703	0.0000	0.0000	0.0000
1.00	2112	0.8826	0.8689	0.2727	0.0023	0.0045
1.20	2112	0.8829	0.8622	0.3333	0.0015	0.0030
1.40	2112	0.8831	0.8625	0.5714	0.0030	0.0060
1.60	2112	0.8822	0.8649	0.1538	0.0015	0.0030
1.80	2112	0.8830	0.8670	0.3333	0.0008	0.0015
2.00	2112	0.8827	0.8655	0.1667	0.0008	0.0015

Ellipsoid quantile. Table 32 varies the ellipsoid quantile q while keeping all other settings fixed. The attack remains stable across the tested quantiles, suggesting that the default $q = 0.95$ does not require fine tuning in this setting.

Categorical and continuous features. Table 33 compares the full attack with a continuous-only variant. The full attack is stronger on datasets with encoded categorical features, while BCW is tied because all 30 features are continuous.

Partial-access attacker. Table 34 evaluates attackers that can inspect only a subset of the training pool. The attack becomes stronger as visibility increases, but it remains nontrivial even at 25%–50% visibility.

Surrogate mismatch. Tables 35 and 36 evaluate mismatch between the surrogate and the victim. The attack remains effective across different surrogate capacities and broader surrogate hyperparameter configurations, although the transfer gap is dataset-dependent.

Transfer across victim model classes. Table 37 evaluates transfer across several tree-based victim models. The attack remains effective across these tree-ensemble models, while the magnitude varies by dataset and victim class.

Paired embedding coordinates. Table 38 reports representative clean/poison coordinate pairs from the same two-dimensional t-SNE embedding. The paired points remain in similar local regions of the embedding, supporting the distributional consistency evidence in Appendix C.1.

Larger-scale runtime. Table 8 reports runtime on larger datasets, including HCDR, a real-world credit-risk benchmark. Runtime increases with dataset scale, but the attack remains practical for the tested settings because the optimization stage can be parallelized across poisoning candidates.

Table 21: CREDIT-G ((w/o. EP)): Single run ε sweep results.

ε	K_{rep}	ACC	AUC	Prec	Rec	F1
Clean	0	0.7520	0.7618	0.6066	0.4933	0.5441
0.20	60	0.7400	0.7469	0.5962	0.4133	0.4882
0.40	60	0.7280	0.7208	0.5814	0.3333	0.4237
0.60	60	0.7320	0.7198	0.6000	0.3200	0.4174
0.80	60	0.7360	0.7441	0.6216	0.3067	0.4107
1.00	60	0.7400	0.7376	0.6667	0.2667	0.3810
1.20	60	0.7280	0.7477	0.6296	0.2267	0.3333
1.40	60	0.7440	0.7555	0.7200	0.2400	0.3600
1.60	60	0.7520	0.7822	0.7407	0.2667	0.3922
1.80	60	0.7160	0.7319	0.5714	0.2133	0.3107
2.00	60	0.7440	0.7414	0.6774	0.2800	0.3962

Table 22: CREDIT-G ((w. EP)): Single run ε sweep results.

ε	K_{rep}	ACC	AUC	Prec	Rec	F1
Clean	0	0.7520	0.7618	0.6066	0.4933	0.5441
0.20	60	0.7400	0.7509	0.6087	0.3733	0.4628
0.40	60	0.7240	0.7205	0.5750	0.3067	0.4000
0.60	60	0.7280	0.7139	0.5946	0.2933	0.3929
0.80	60	0.7080	0.7438	0.5333	0.2133	0.3048
1.00	60	0.7160	0.7483	0.5714	0.2133	0.3107
1.20	60	0.7160	0.7414	0.5588	0.2533	0.3486
1.40	60	0.7440	0.7716	0.6571	0.3067	0.4182
1.60	60	0.7480	0.7989	0.6765	0.3067	0.4220
1.80	60	0.7640	0.7598	0.8636	0.2533	0.3918
2.00	60	0.7320	0.7541	0.6333	0.2533	0.3619

Filtering and training-time defenses. Table 39 reports detection rates for several filtering baselines, together with false-positive rates on clean data. Table 40 reports attack effectiveness after filtering and after Deep Partition Aggregation (DPA). These results broaden the stealthiness evaluation beyond the supervised detectors in Table 6, while the scope remains limited to the tested defenses.

Table 23: BCW ((w/o. EP)): Single run ε sweep results.

ε	K_{rep}	ACC	AUC	Prec	Rec	F1
Clean	0	0.9720	0.9933	1.0000	0.9245	0.9608
0.20	34	0.9790	0.9973	1.0000	0.9434	0.9709
0.40	34	0.9580	0.9937	0.9796	0.9057	0.9412
0.60	34	0.9510	0.9895	0.9600	0.9057	0.9320
0.80	34	0.9510	0.9910	0.9792	0.8868	0.9307
1.00	34	0.9510	0.9918	0.9600	0.9057	0.9320
1.20	34	0.9510	0.9889	0.9600	0.9057	0.9320
1.40	34	0.9580	0.9918	0.9608	0.9245	0.9423
1.60	34	0.9510	0.9906	0.9600	0.9057	0.9320
1.80	34	0.9650	0.9937	0.9615	0.9434	0.9524
2.00	34	0.9580	0.9929	0.9608	0.9245	0.9423

Table 24: BCW ((w. EP)): Single run ε sweep results.

ε	K_{rep}	ACC	AUC	Prec	Rec	F1
Clean	0	0.9720	0.9933	1.0000	0.9245	0.9608
0.20	34	0.9790	0.9960	1.0000	0.9434	0.9709
0.40	34	0.9371	0.9933	0.9783	0.8491	0.9091
0.60	34	0.9580	0.9893	0.9796	0.9057	0.9412
0.80	34	0.9441	0.9901	0.9787	0.8679	0.9200
1.00	34	0.9650	0.9895	1.0000	0.9057	0.9505
1.20	34	0.9510	0.9920	0.9600	0.9057	0.9320
1.40	34	0.9580	0.9927	0.9608	0.9245	0.9423
1.60	34	0.9650	0.9925	0.9800	0.9245	0.9515
1.80	34	0.9720	0.9943	0.9804	0.9434	0.9615
2.00	34	0.9650	0.9956	0.9615	0.9434	0.9524

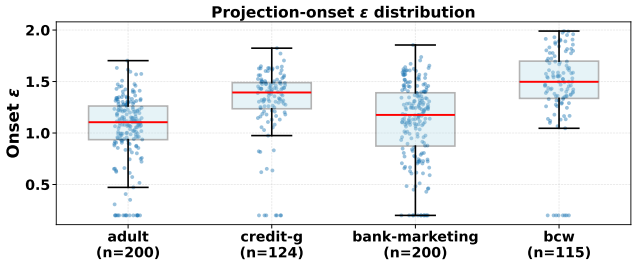

Figure 8: **Projection onset distribution across datasets.** Each point is one candidate; the onset value is the smallest ε at which the ellipsoid projection is triggered at least once during the poisoning procedure (including update iterates and FD probe points).

Table 25: AMAZON: Single run ε sweep (target class 0).

	(w. EP)					(w/o. EP)				
ε	ACC	AUC	Prec_0	Rec_0	F1_0	ACC	AUC	Prec_0	Rec_0	F1_0
Clean	0.9437	0.8023	0.8125	0.0343	0.0658	0.9437	0.8023	0.8125	0.0343	0.0658
0.20	0.9428	0.7802	0.6250	0.0264	0.0506	0.9431	0.7860	0.6667	0.0317	0.0605
0.40	0.9435	0.7997	0.7368	0.0369	0.0704	0.9435	0.7983	0.7368	0.0369	0.0704
0.60	0.9425	0.7920	0.7500	0.0079	0.0157	0.9422	0.7947	0.5000	0.0079	0.0156
0.80	0.9431	0.7908	0.7143	0.0264	0.0509	0.9425	0.7899	0.6667	0.0106	0.0208
1.00	0.9439	0.7869	0.7895	0.0396	0.0754	0.9429	0.7942	0.7273	0.0211	0.0410
1.20	0.9422	0.7931	0.5000	0.0053	0.0104	0.9428	0.8014	0.7500	0.0158	0.0310
1.40	0.9426	0.7953	0.6000	0.0237	0.0457	0.9428	0.7940	0.6429	0.0237	0.0458
1.60	0.9429	0.7986	0.7778	0.0185	0.0361	0.9429	0.7886	0.6923	0.0237	0.0459
1.80	0.9425	0.7976	0.6000	0.0158	0.0308	0.9429	0.7955	0.8571	0.0158	0.0311
2.00	0.9423	0.7888	0.5714	0.0106	0.0207	0.9425	0.7905	0.6250	0.0132	0.0258

Table 26: HIGGS: Single run ε sweep comparison.

	(w. EP)					(w/o. EP)				
ε	**ACC**	**AUC**	**Prec**	**Rec**	**F1**	**ACC**	**AUC**	**Prec**	**Rec**	**F1**
Clean	0.7137	0.7916	0.6969	0.6951	0.6960	0.7137	0.7916	0.6969	0.6951	0.6960
0.20	0.7157	0.7921	0.6980	0.6997	0.6989	0.7167	0.7924	0.6995	0.6995	0.6995
0.60	0.7144	0.7914	0.6977	0.6957	0.6967	0.7163	0.7924	0.7008	0.6949	0.6978
0.80	0.7140	0.7910	0.6973	0.6951	0.6962	0.7146	0.7912	0.6991	0.6929	0.6960
1.00	0.7157	0.7916	0.6998	0.6950	0.6974	0.7151	0.7912	0.6988	0.6954	0.6971
1.20	0.7147	0.7919	0.6976	0.6969	0.6973	0.7137	0.7915	0.6958	0.6978	0.6968
1.40	0.7148	0.7916	0.6979	0.6968	0.6973	0.7153	0.7909	0.6997	0.6939	0.6968
1.60	0.7157	0.7922	0.6999	0.6949	0.6974	0.7135	0.7911	0.6962	0.6959	0.6961
1.80	0.7136	0.7912	0.6987	0.6899	0.6943	0.7129	0.7914	0.6978	0.6895	0.6936
2.00	0.7157	0.7917	0.7008	0.6928	0.6968	0.7157	0.7918	0.6995	0.6958	0.6977

Table 27: HELOC: Single run ε sweep comparison.

	(w. EP)					(w/o. EP)				
ε	**ACC**	**AUC**	**Prec**	**Rec**	**F1**	**ACC**	**AUC**	**Prec**	**Rec**	**F1**
Clean	0.7235	0.7981	0.7118	0.7510	0.7309	0.7235	0.7981	0.7118	0.7510	0.7309
0.20	0.7220	0.7914	0.7102	0.7500	0.7296	0.7235	0.7946	0.7079	0.7610	0.7335
0.40	0.7230	0.7916	0.7057	0.7650	0.7342	0.7175	0.7902	0.7058	0.7460	0.7253
0.60	0.7190	0.7888	0.7051	0.7530	0.7282	0.7190	0.7915	0.7043	0.7550	0.7288
0.80	0.7220	0.7917	0.7110	0.7480	0.7290	0.7135	0.7901	0.7031	0.7390	0.7206
1.00	0.7110	0.7879	0.7033	0.7300	0.7164	0.7190	0.7920	0.7135	0.7320	0.7226
1.20	0.7190	0.7904	0.7173	0.7230	0.7201	0.7170	0.7884	0.7136	0.7250	0.7192
1.40	0.7160	0.7909	0.7122	0.7250	0.7185	0.7190	0.7926	0.7181	0.7210	0.7196
1.60	0.7180	0.7911	0.7158	0.7230	0.7194	0.7180	0.7881	0.7176	0.7190	0.7183
1.80	0.7255	0.7951	0.7204	0.7370	0.7286	0.7165	0.7918	0.7159	0.7180	0.7169
2.00	0.7235	0.7886	0.7211	0.7290	0.7250	0.7220	0.7916	0.7238	0.7180	0.7209

Table 28: **Attack effectiveness vs. tree complexity** ($\varepsilon = 1.0$, $\rho = 10\%$). Entries show Recall. Δ denotes degradation (Clean $-$ Poisoned).

Dataset	**num_leaves**	**Clean**	**Poisoned**	**Δ**	**Δ%**
Adult	15	0.636	0.299	-0.337	-53%
	31	0.641	0.335	-0.306	-48%
	63	0.650	0.399	-0.251	-39%
	127	0.646	0.445	-0.201	-31%
Credit-G	15	0.493	0.267	-0.226	-46%
	31	0.493	0.267	-0.226	-46%
	63	0.493	0.267	-0.226	-46%
	127	0.493	0.253	-0.240	-49%

Table 29: Finite-difference step-size sensitivity. Entries are Recall (%) under different perturbation budgets.

Dataset	**FD scale**	$\varepsilon = 0.5$	$\varepsilon = 1.0$	$\varepsilon = 1.5$	$\varepsilon = 2.0$
Adult	0.1	38.3	37.9	31.7	40.6
Adult	0.3	36.1	30.6	36.7	41.8
Adult	0.7	32.5	27.3	29.8	34.1
Credit-G	0.1	37.3	30.7	36.0	36.0
Credit-G	0.3	36.0	26.7	34.7	37.3
Credit-G	0.7	25.3	24.0	20.0	21.3

Table 30: Learning-rate ablation. Entries are Recall (%) under different perturbation budgets.

Dataset	**LR**	$\varepsilon = 0.5$	$\varepsilon = 1.0$	$\varepsilon = 1.5$	$\varepsilon = 2.0$
Adult	0.05	35.7	29.3	31.2	33.4
Adult	0.10	36.1	30.6	36.7	41.8
Adult	0.30	34.7	31.9	36.7	43.5
Credit-G	0.05	30.7	33.3	21.3	21.3
Credit-G	0.10	36.0	26.7	34.7	37.3
Credit-G	0.30	36.0	41.3	24.0	21.3

Table 31: Comparison of plausibility constraints. Entries are Recall drop (%).

Dataset	Per-feature clipping	ℓ_2-ball	EP (Ours)
Adult	58.0	51.9	59.3
Credit-G	53.8	64.1	66.7
BCW	9.8	5.9	9.8

Table 32: Sensitivity to ellipsoid quantile q. Entries are Recall drop (%).

Dataset	q	Recall drop (%)
Adult	0.90	49.0
Adult	0.95	49.8
Adult	0.98	49.2
Credit-G	0.90	54.1
Credit-G	0.95	54.1
Credit-G	0.98	48.6

Table 33: Full attack vs. continuous-only attack. Entries are Recall drop (%).

Dataset	Continuous-only	Full (all features)	Active dims
Adult	55.5	60.6	6/14
Credit-G	37.5	56.3	7/20
BCW	10.6	10.6	30/30

Table 34: Partial-access attacker. Entries are Recall drop (%).

Visible data	Adult	Credit-G
25%	20.4	27.0
50%	22.1	35.1
75%	32.3	43.2
100%	49.8	59.5

Table 35: Surrogate mismatch under different surrogate numbers of leaves.

Dataset	Surrogate num_leaves	Recall drop (%)	F1 drop (%)
Adult	31	57.0	42.0
Adult	63	49.8	34.5
Adult	127	43.4	28.8
Credit-G	31	54.1	39.3
Credit-G	63	54.1	39.3
Credit-G	127	54.1	39.3

Table 36: Surrogate training mismatch for a fixed LightGBM victim.

Dataset	LR	num_leaves	min_data_in_leaf	reg_lambda	n_estimators	Recall drop (%)	F1 drop (%)
Adult	0.05	31	100	1.0	40	53.4	37.1
Adult	0.10	63	50	1.0	80	49.8	34.5
Adult	0.20	127	20	1.0	120	32.9	20.4
Adult	0.10	63	100	5.0	80	57.5	42.4
Credit-G	0.05	31	100	1.0	40	48.6	34.1
Credit-G	0.10	63	50	1.0	80	54.1	39.3
Credit-G	0.20	127	20	1.0	120	35.1	16.8
Credit-G	0.10	63	100	5.0	80	62.2	48.0

Table 37: Transfer across multiple tree-based victim models.

Dataset	Victim model	Recall drop (%)	F1 drop (%)
Adult	LightGBM	49.8	34.5
Adult	Random Forest	46.1	32.1
Adult	Extra Trees	48.8	34.9
Adult	sklearn GBDT	46.5	31.8
Adult	HistGBDT	49.1	33.5
Credit-G	LightGBM	54.1	39.3
Credit-G	Random Forest	70.3	59.4
Credit-G	Extra Trees	57.1	43.3
Credit-G	sklearn GBDT	58.3	47.8
Credit-G	HistGBDT	33.3	26.1

Table 38: Representative t-SNE coordinates of paired clean and poisoned samples.

Dataset	Clean (x, y)	Poison (x, y)
Adult	$(-21.62, -13.73)$	$(-21.62, -13.73)$
Adult	$(-36.87, 5.24)$	$(-36.86, 5.25)$
Adult	$(-40.06, -3.02)$	$(-40.06, -2.99)$
Adult	$(-13.28, -11.09)$	$(-13.24, -11.08)$
Credit-G	$(23.36, -14.63)$	$(22.79, -14.35)$
Credit-G	$(-17.20, 10.54)$	$(-17.21, 11.19)$
Credit-G	$(9.12, -19.96)$	$(8.65, -20.42)$
Credit-G	$(12.27, -8.46)$	$(12.79, -8.04)$

Table 39: Filtering-based detection. Detection rate is measured on poisoned samples; false positive rate is measured on clean samples.

Dataset/metric	Isolation Forest	LOF	Robust Cov.	K-NN
Adult detection rate	13.8%	17.6%	1.3%	8.9%
Adult false positive	10.0%	8.5%	10.0%	10.1%
Credit-G detection rate	11.7%	13.3%	10.0%	8.3%
Credit-G false positive	10.0%	8.5%	10.0%	10.3%

Table 40: Attack effectiveness under filtering and training-time defenses.

Dataset	Defense	Recall drop (%)	F1 drop (%)
Adult	None	58.1	43.3
Adult	Isolation Forest	56.2	41.8
Adult	LOF	55.7	41.0
Adult	Robust Cov.	52.8	38.1
Adult	K-NN	55.2	40.6
Adult	DPA	43.1	27.6
Credit-G	None	62.2	49.1
Credit-G	Isolation Forest	58.8	45.9
Credit-G	LOF	65.7	54.3
Credit-G	Robust Cov.	83.8	74.8
Credit-G	K-NN	66.7	54.4
Credit-G	DPA	53.8	40.6

D Additional Statistical Results

D.1 Preprocessing Transfer and Categorical Validity

The attack transfers across independently fitted victim encoders that are unavailable to the attacker (Table 41). Moreover, categorical profile mapping retains at least 95.2% of F1 and Recall degradation (Table 5). The optimized categorical block is mapped to its nearest joint profile observed in clean training data using squared standardized Euclidean distance. Numerical coordinates and labels remain unchanged during mapping. The resulting raw record is then passed through the victim's preprocessing pipeline. Each paired run retrains the victim before and after mapping with all other choices fixed.

Table 41: Transfer to independently fitted victim encoders after mapping poisons back to raw categorical values. F1 and Recall drops (%) are mean ± standard deviation.

Data	Victim encoding	F1 drop (%)	Recall drop (%)
Adult	One-hot	22.45 ± 4.53	36.76 ± 5.31
Adult	Ordinal	23.46 ± 4.52	37.74 ± 5.26
Credit-G	One-hot	11.12 ± 10.48	19.93 ± 12.98
Credit-G	Ordinal	18.19 ± 13.31	27.13 ± 16.26

D.2 Budget Dependence across Random Seeds

Table 42: F1 and Recall across perturbation budgets, reported as mean ± standard deviation over 200 random seeds. Lower scores indicate stronger attacks.

ε	Adult F1	Adult Recall	Credit-G F1	Credit-G Recall	BCW F1	BCW Recall
0.0	0.7107 ± 0.0059	0.6544 ± 0.0079	0.4983 ± 0.0462	0.4304 ± 0.0546	0.9467 ± 0.0208	0.9375 ± 0.0291
0.2	0.4635 ± 0.0225	0.3205 ± 0.0209	0.4184 ± 0.0519	0.3212 ± 0.0533	0.9514 ± 0.0191	0.9515 ± 0.0285
0.4	0.4465 ± 0.0285	0.3086 ± 0.0253	0.3823 ± 0.0502	0.2863 ± 0.0478	0.9357 ± 0.0229	0.9213 ± 0.0356
0.6	0.4519 ± 0.0228	0.3143 ± 0.0198	0.3624 ± 0.0564	0.2609 ± 0.0510	0.9254 ± 0.0275	0.8950 ± 0.0419
0.8	0.4555 ± 0.0221	0.3214 ± 0.0209	0.3375 ± 0.0614	0.2365 ± 0.0526	0.9260 ± 0.0274	0.8968 ± 0.0449
1.0	0.4780 ± 0.0209	0.3485 ± 0.0208	0.3165 ± 0.0632	0.2176 ± 0.0527	0.9299 ± 0.0238	0.9123 ± 0.0364
1.2	0.4483 ± 0.0253	0.3186 ± 0.0245	0.3073 ± 0.0601	0.2090 ± 0.0499	0.9319 ± 0.0231	0.9211 ± 0.0338
1.4	0.4403 ± 0.0285	0.3102 ± 0.0276	0.3169 ± 0.0665	0.2172 ± 0.0544	0.9325 ± 0.0239	0.9258 ± 0.0343
1.6	0.4587 ± 0.0403	0.3282 ± 0.0370	0.3148 ± 0.0683	0.2137 ± 0.0558	0.9336 ± 0.0236	0.9282 ± 0.0349
1.8	0.5003 ± 0.0342	0.3689 ± 0.0333	0.3088 ± 0.0644	0.2087 ± 0.0534	0.9335 ± 0.0239	0.9286 ± 0.0352
2.0	0.5081 ± 0.0349	0.3792 ± 0.0345	0.3535 ± 0.0656	0.2477 ± 0.0559	0.9348 ± 0.0245	0.9318 ± 0.0349

Table 43: Paired score differences at $\varepsilon = 2.0$ relative to $\varepsilon_0 = 1.6$ (Adult), 1.8 (Credit-G), and 0.8 (BCW), over 200 matched seeds. Positive values indicate weaker attacks at the larger budget. Brackets give 95% confidence intervals for two-sided tests.

Dataset	Metric	Difference [95% CI]	t	p
Adult	F1	$0.0493[0.0438, 0.0549]$	17.4989	< 0.0001
Adult	Recall	$0.0510[0.0457, 0.0564]$	18.7508	< 0.0001
Credit-G	F1	$0.0448[0.0360, 0.0535]$	10.0726	< 0.0001
Credit-G	Recall	$0.0390[0.0317, 0.0463]$	10.5046	< 0.0001
BCW	F1	$0.0088[0.0055, 0.0121]$	5.1978	< 0.0001
BCW	Recall	$0.0350[0.0296, 0.0404]$	12.8858	< 0.0001

To keep the absolute update magnitude constant, we use $\eta(\varepsilon) = 0.7\varepsilon_0/\varepsilon$ with the same data splits, candidates, models, and two update iterations. The reference budgets ε_0 are 1.6 for Adult, 1.8 for Credit-G, and 0.8 for BCW.

Table 44: Paired score differences at $\varepsilon = 2.0$ relative to ε_0 with constant update magnitude, over 200 matched seeds. Brackets give 95% confidence intervals for two-sided tests.

Dataset	Metric	Difference [95% CI]	t	p
Adult	F1	$0.057430[0.051725, 0.063136]$	19.8495	< 0.0001
Adult	Recall	$0.059887[0.054359, 0.065415]$	21.3616	< 0.0001
Credit-G	F1	$0.045517[0.036961, 0.054073]$	10.4904	< 0.0001
Credit-G	Recall	$0.039267[0.032240, 0.046293]$	11.0201	< 0.0001
BCW	F1	$0.006699[0.003319, 0.010079]$	3.9086	0.00013
BCW	Recall	$0.037547[0.032359, 0.042736]$	14.2702	< 0.0001

D.3 Ellipsoidal Projection across Budgets

Table 45: F1 with and without EP across 200 matched seeds. Entries are mean $\pm$ standard deviation, with lower scores indicating stronger attacks.

ε	Adult w/o. EP	Adult w. EP	Credit-G w/o. EP	Credit-G w. EP
0.0	0.7107 ± 0.0059	0.7107 ± 0.0059	0.4983 ± 0.0462	0.4983 ± 0.0462
0.2	0.4635 ± 0.0225	0.4633 ± 0.0222	0.4184 ± 0.0519	0.4108 ± 0.0535
0.4	0.4465 ± 0.0285	0.4412 ± 0.0251	0.3823 ± 0.0502	0.3775 ± 0.0511
0.6	0.4519 ± 0.0228	0.4151 ± 0.0218	0.3624 ± 0.0564	0.3381 ± 0.0547
0.8	0.4555 ± 0.0221	0.4338 ± 0.0405	0.3375 ± 0.0614	0.3141 ± 0.0641
1.0	0.4780 ± 0.0209	0.4162 ± 0.0183	0.3165 ± 0.0632	0.3115 ± 0.0646
1.2	0.4483 ± 0.0253	0.4370 ± 0.0296	0.3073 ± 0.0601	0.3133 ± 0.0647
1.4	0.4403 ± 0.0285	0.4361 ± 0.0260	0.3169 ± 0.0665	0.3095 ± 0.0629
1.6	0.4587 ± 0.0403	0.4368 ± 0.0321	0.3148 ± 0.0683	0.3093 ± 0.0596
1.8	0.5003 ± 0.0342	0.4333 ± 0.0275	0.3088 ± 0.0644	0.3044 ± 0.0657
2.0	0.5081 ± 0.0349	0.4372 ± 0.0382	0.3535 ± 0.0656	0.3205 ± 0.0655

Table 46: Recall with and without EP across 200 matched seeds. Entries are mean $\pm$ standard deviation, with lower scores indicating stronger attacks.

ε	Adult w/o. EP	Adult w. EP	Credit-G w/o. EP	Credit-G w. EP
0.0	0.6544 ± 0.0079	0.6544 ± 0.0079	0.4304 ± 0.0546	0.4304 ± 0.0546
0.2	0.3205 ± 0.0209	0.3203 ± 0.0204	0.3212 ± 0.0533	0.3099 ± 0.0541
0.4	0.3086 ± 0.0253	0.3029 ± 0.0222	0.2863 ± 0.0478	0.2779 ± 0.0467
0.6	0.3143 ± 0.0198	0.2824 ± 0.0191	0.2609 ± 0.0510	0.2384 ± 0.0475
0.8	0.3214 ± 0.0209	0.2990 ± 0.0355	0.2365 ± 0.0526	0.2149 ± 0.0534
1.0	0.3485 ± 0.0208	0.2859 ± 0.0167	0.2176 ± 0.0527	0.2126 ± 0.0539
1.2	0.3186 ± 0.0245	0.3041 ± 0.0281	0.2090 ± 0.0499	0.2139 ± 0.0530
1.4	0.3102 ± 0.0276	0.3041 ± 0.0244	0.2172 ± 0.0544	0.2123 ± 0.0527
1.6	0.3282 ± 0.0370	0.3096 ± 0.0293	0.2137 ± 0.0558	0.2103 ± 0.0493
1.8	0.3689 ± 0.0333	0.3015 ± 0.0265	0.2087 ± 0.0534	0.2057 ± 0.0529
2.0	0.3792 ± 0.0345	0.3056 ± 0.0356	0.2477 ± 0.0559	0.2206 ± 0.0553

Table 47: Pointwise EP comparisons on Adult over 200 matched seeds. Differences are w/o. EP minus w. EP, with positive values indicating an EP improvement. Brackets give 95% confidence intervals. Two-sided p values are reported for every tested budget.

ε	Metric	Difference [95% CI]	t	p
0.2	F1	$0.0002[0.0000, 0.0004]$	1.6929	0.0920
0.2	Recall	$0.0003[0.0000, 0.0005]$	2.0741	0.0394
0.4	F1	$0.0053[0.0038, 0.0069]$	6.7248	< 0.0001
0.4	Recall	$0.0057[0.0042, 0.0071]$	7.5414	< 0.0001
0.6	F1	$0.0368[0.0332, 0.0405]$	19.7462	< 0.0001
0.6	Recall	$0.0319[0.0287, 0.0351]$	19.7792	< 0.0001
0.8	F1	$0.0217[0.0161, 0.0273]$	7.6637	< 0.0001
0.8	Recall	$0.0224[0.0174, 0.0275]$	8.8117	< 0.0001
1.0	F1	$0.0618[0.0585, 0.0652]$	36.2096	< 0.0001
1.0	Recall	$0.0626[0.0593, 0.0659]$	37.6518	< 0.0001
1.2	F1	$0.0113[0.0066, 0.0159]$	4.7845	< 0.0001
1.2	Recall	$0.0145[0.0100, 0.0189]$	6.4269	< 0.0001
1.4	F1	$0.0042[-0.0006, 0.0090]$	1.7219	0.0866
1.4	Recall	$0.0060[0.0014, 0.0107]$	2.5734	0.0108
1.6	F1	$0.0220[0.0178, 0.0261]$	10.4535	< 0.0001
1.6	Recall	$0.0186[0.0148, 0.0223]$	9.7331	< 0.0001
1.8	F1	$0.0670[0.0620, 0.0720]$	26.4352	< 0.0001
1.8	Recall	$0.0674[0.0625, 0.0723]$	27.0752	< 0.0001
2.0	F1	$0.0708[0.0644, 0.0772]$	21.8808	< 0.0001
2.0	Recall	$0.0736[0.0675, 0.0798]$	23.6761	< 0.0001

Table 48: Pointwise EP comparisons on Credit-G over 200 matched seeds. Differences are w/o. EP minus w. EP, with positive values indicating an EP improvement. Brackets give 95% confidence intervals. Two-sided p values are reported for every tested budget.

ε	Metric	Difference [95% CI]	t	p
0.2	F1	$0.0076[0.0016, 0.0135]$	2.5061	0.0130
0.2	Recall	$0.0113[0.0056, 0.0171]$	3.9011	0.0001
0.4	F1	$0.0047[-0.0017, 0.0112]$	1.4443	0.1502
0.4	Recall	$0.0084[0.0026, 0.0142]$	2.8664	0.0046
0.6	F1	$0.0244[0.0178, 0.0310]$	7.2986	< 0.0001
0.6	Recall	$0.0225[0.0170, 0.0279]$	8.1743	< 0.0001
0.8	F1	$0.0234[0.0153, 0.0316]$	5.6642	< 0.0001
0.8	Recall	$0.0217[0.0150, 0.0283]$	6.4441	< 0.0001
1.0	F1	$0.0049[-0.0017, 0.0116]$	1.4567	0.1468
1.0	Recall	$0.0050[-0.0004, 0.0104]$	1.8172	0.0707
1.2	F1	$-0.0060[-0.0140, 0.0021]$	-1.4621	0.1453
1.2	Recall	$-0.0049[-0.0115, 0.0017]$	-1.4562	0.1469
1.4	F1	$0.0074[-0.0004, 0.0152]$	1.8698	0.0630
1.4	Recall	$0.0049[-0.0013, 0.0112]$	1.5587	0.1207
1.6	F1	$0.0055[-0.0032, 0.0142]$	1.2405	0.2162
1.6	Recall	$0.0034[-0.0038, 0.0106]$	0.9354	0.3507
1.8	F1	$0.0044[-0.0031, 0.0119]$	1.1606	0.2472
1.8	Recall	$0.0031[-0.0030, 0.0092]$	0.9930	0.3219
2.0	F1	$0.0331[0.0236, 0.0426]$	6.8674	< 0.0001
2.0	Recall	$0.0271[0.0192, 0.0350]$	6.7826	< 0.0001

Table 49: EP effect averaged over the ten nonzero budgets within each of 200 seeds. Differences are w/o. EP minus w. EP, with 95% confidence intervals and two-sided t-tests on the seed averages.

Dataset	Metric	Mean difference [95% CI]	t	p
Adult	F1	$0.0301[0.0284, 0.0319]$	33.7766	< 0.0001
Adult	Recall	$0.0303[0.0286, 0.0320]$	35.9060	< 0.0001
Credit-G	F1	$0.0109[0.0083, 0.0136]$	8.2087	< 0.0001
Credit-G	Recall	$0.0103[0.0081, 0.0124]$	9.2993	< 0.0001

D.4 Expanded Baseline Comparisons

Table 50: Baseline comparisons over 200 matched seeds. Clean rows give test scores, whereas other rows give poisoned minus clean scores. Entries are mean $\pm$ standard deviation.

Dataset	Method	ACC	AUC	F1	Recall
Adult	Clean	0.8725 ± 0.0025	0.9277 ± 0.0020	0.7107 ± 0.0059	0.6544 ± 0.0079
Adult	Random Noise	-0.0142 ± 0.0027	-0.0029 ± 0.0007	-0.0956 ± 0.0093	-0.1811 ± 0.0110
Adult	NN-Transfer	-0.0147 ± 0.0027	-0.0033 ± 0.0008	-0.0981 ± 0.0091	-0.1845 ± 0.0108
Adult	Min-Max	-0.0148 ± 0.0027	-0.0034 ± 0.0009	-0.0987 ± 0.0092	-0.1852 ± 0.0109
Adult	Ours (w. EP)	-0.0542 ± 0.0071	-0.0507 ± 0.0129	-0.2500 ± 0.0312	-0.3294 ± 0.0297
Credit-G	Clean	0.7411 ± 0.0222	0.7551 ± 0.0288	0.4983 ± 0.0462	0.4304 ± 0.0546
Credit-G	Random Noise	-0.0093 ± 0.0223	-0.0103 ± 0.0180	-0.0835 ± 0.0498	-0.1114 ± 0.0491
Credit-G	NN-Transfer	-0.0082 ± 0.0216	-0.0099 ± 0.0193	-0.0945 ± 0.0544	-0.1265 ± 0.0539
Credit-G	Min-Max	-0.0063 ± 0.0225	-0.0094 ± 0.0196	-0.0887 ± 0.0565	-0.1211 ± 0.0551
Credit-G	Ours (w. EP)	-0.0187 ± 0.0235	-0.0402 ± 0.0318	-0.1758 ± 0.0651	-0.2077 ± 0.0602

Table 51: Paired comparisons against Min-Max over 200 matched seeds. Positive additional degradation favors our method. Brackets give 95% confidence intervals, and p values are from two-sided paired t-tests.

Dataset	Metric	Additional degradation [95% CI]	t	p
Adult	ACC	$0.0393[0.0384, 0.0403]$	79.9418	< 0.0001
Adult	AUC	$0.0473[0.0455, 0.0491]$	51.8070	< 0.0001
Adult	F1	$0.1514[0.1470, 0.1557]$	68.1143	< 0.0001
Adult	Recall	$0.1442[0.1399, 0.1484]$	66.9497	< 0.0001
Credit-G	ACC	$0.0124[0.0095, 0.0153]$	8.4198	< 0.0001
Credit-G	AUC	$0.0308[0.0264, 0.0352]$	13.7402	< 0.0001
Credit-G	F1	$0.0871[0.0779, 0.0963]$	18.6898	< 0.0001
Credit-G	Recall	$0.0866[0.0785, 0.0947]$	20.9812	< 0.0001

E Impact Statement

This paper presents a study on the vulnerabilities of Gradient Boosting Decision Trees (GBDTs) to stealthy clean-label poisoning attacks. Given the ubiquity of GBDTs in high-stakes tabular domains—such as finance, healthcare, and fraud detection—understanding their failure modes is critical for system reliability. While our work introduces a more potent attack vector that exploits statistical plausibility constraints, our primary goal is to expose the limitations of current outlier-based defenses and motivate the development of robust training algorithms for tree ensembles. We believe that identifying these stealthy vulnerabilities is a necessary step toward building secure machine learning systems that can withstand adaptive adversaries in real-world deployments.

F Scope

The paper studies clean-label availability poisoning for GBDT-style tabular classifiers. The attacker perturbs selected training features while preserving labels, and the victim is evaluated on an untouched clean test set. Our empirical claims are scoped to the public tabular benchmarks, poisoning budgets, perturbation constraints, and GBDT implementations evaluated in the paper and appendix.

G Limitations

The strongest setting assumes that the attacker can inspect the training distribution, fit a surrogate model, and select poisoning candidates from the training pool. This is a standard setting for controlled clean-label poisoning evaluation, but it is stronger than many deployed scenarios. We therefore additionally evaluate partial-access and surrogate-mismatch settings in Appendix C.4. The ellipsoidal constraint is a distributional plausibility constraint in the encoded feature space; it does not by itself guarantee semantic validity for every possible real-world categorical domain. The defense evaluation is also limited to filtering and aggregation-style baselines, so the results should be interpreted as evidence of vulnerability under the tested defenses rather than as a claim of robustness against all possible training-time defenses. A complete formal characterization of non-monotonicity in tree-ensemble poisoning remains an interesting direction for future work.

H Broader Impacts

This work studies a training-time vulnerability of GBDT models in tabular learning. The positive impact is to identify failure modes in models that are commonly used in security- and risk-sensitive settings, thereby motivating stronger data validation, robust training, and poisoning defenses for tree ensembles. The negative impact is dual-use: the attack procedure could be misused by an adversary who can modify training data. We frame the work as a controlled security analysis on public benchmarks and include stealthiness and filtering analyses to help characterize the limits of simple defenses.

I Responsible Use

The released materials are intended to support reproducibility and defensive research rather than deployment against real systems. The experiments are conducted on public benchmark datasets under a clearly stated threat model. The paper reports detection and filtering analyses so that the same empirical setup can inform future work on defenses and data auditing for tree-based models.

J Responsible Release and Safeguards

The work does not release a pretrained model, scraped dataset, generative model, or human-subject dataset. The supplementary package contains code for reproducing the reported experiments on public benchmark datasets and records data access instructions and license or terms-of-use information. Because the main release is research code for public benchmark experiments rather than a high-risk model or dataset, no model-access safeguard is required.

K Human Subjects

No crowdsourcing or human-subject experiments were conducted. The study uses public tabular benchmark datasets and does not collect new data from participants. Accordingly, no IRB-style approval or participant compensation procedure was required for the reported experiments.

L LLM Use

LLMs were not used as an important, original, or non-standard component of the research method. The proposed attack, experiments, and scientific conclusions do not depend on LLM-generated components. Any language-model assistance, if used, was limited to writing, editing, or formatting support and did not affect the scientific methodology or experimental results.

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: The abstract and introduction state the main contributions, including the clean-label threat model for GBDTs, the non-monotonic ε behavior, and the ellipsoidal projection method. The experimental sections report results that support these claims within the stated tabular-data setting.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The paper states the main threat-model assumptions in Section 3.1, the experimental scope in Section 4, and additional limitations in Appendix G. Compute resources are reported in Appendix A.8.

   Guidelines:

   - The answer [N/A] 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: [N/A]

Justification: The paper does not present formal theoretical results such as theorems or lemmas. The mathematical content consists of the attack formulation, finite-difference direction estimation, and projection operation.

Guidelines:

- The answer [N/A] 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: The paper provides the dataset setup, victim model configuration, attack configuration, metrics, and additional reproducibility details in Section 4 and Appendix A. Hyperparameters and data splits are further summarized in the appendix tables.

Guidelines:

- The answer [N/A] 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: [Yes]

   Justification: The experiments use publicly available benchmark datasets, and an anonymized code package with scripts and reproduction instructions will be included with the submission. The paper also provides dataset details, hyperparameters, and experimental protocols in Appendix A.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setup is described in Section 4.1, including datasets, victim model, attack configuration, and evaluation metrics. Appendix A provides additional dataset splits, LightGBM settings, and attack hyperparameters.

   Guidelines:

   - The answer [N/A] 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: [Yes]

Justification: The main experimental curves report mean±std over five seeds, and the detection analysis reports cross-validation AUC with standard deviations. The captions and surrounding text specify when results are averaged over multiple seeds.

Guidelines:

- The answer [N/A] 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: Appendix A.8 reports the CPU-only setting, Intel Xeon Gold 6346 processor, single-threaded execution, and wall-clock runtime reporting protocol. Additional large-scale runtime results are provided in Table 8.

   Guidelines:

   - The answer [N/A] 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: The work uses standard tabular benchmarks and studies a security vulnerability under a clearly stated threat model. The paper discusses the dual-use nature of the attack and its motivation for improving robust training in the Impact Statement.

   Guidelines:

   - The answer [N/A] 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: [Yes]

    Justification: The Impact Statement discusses the positive impact of identifying vulnerabilities in high-stakes tabular systems and the negative dual-use risk of presenting a stronger clean-label poisoning attack. It also frames the work as motivation for improved defenses.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk pretrained models, scraped datasets, or generative models. The main risk is dual-use attack knowledge, which is discussed in the Impact Statement.

    Guidelines:

    - The answer [N/A] 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: [Yes]

Justification: The paper uses publicly available benchmark datasets and standard software packages, and the original dataset and software sources are credited through citations and experimental descriptions. The supplemental code package will include data access instructions and license/terms information, and all assets are used under their intended research terms.

Guidelines:

- The answer [N/A] 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: [Yes]

Justification: The paper does not introduce a new dataset or model checkpoint, but it will release an anonymized code package for the proposed attack and experiments. The code package will include documentation, reproduction instructions, and data preparation details.

Guidelines:

- The answer [N/A] 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: [N/A]

Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

Guidelines:

- The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects, so IRB approval or equivalent review is not applicable.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development and experiments do not involve LLMs as an important, original, or non-standard component. Any LLM use for writing or formatting, if any, does not affect the methodology or results.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.