The Effect of Data Poisoning on Counterfactual Explanations

André Artelt
Faculty of Technology
Bielefeld University
Bielefeld, Germany
aartelt@techfak.uni-bielefeld.de

Shubham Sharma
J.P. Morgan AI Research
New York City, USA
shubham.x2.sharma@jpmchase.com

Freddy Lecué
Inria, Sophia Antipolis
France
freddy.lecue@inria.fr

Barbara Hammer
Faculty of Technology
Bielefeld University
Bielefeld, Germany
bhammer@techfak.uni-bielefeld.de

Abstract

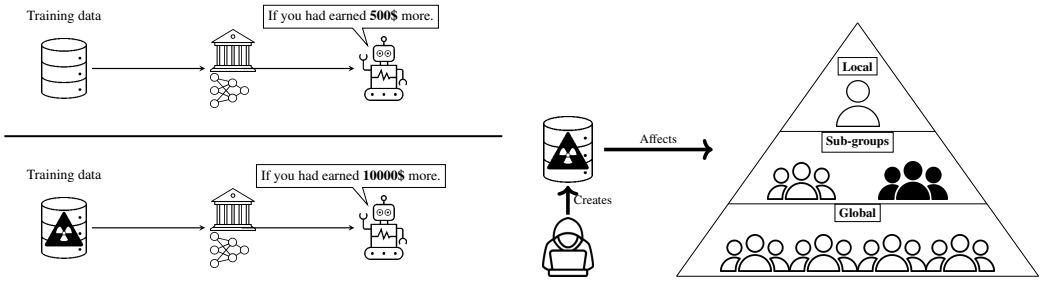
Counterfactual explanations provide a popular method for analyzing the predictions of black-box systems, and they can offer the opportunity for computational recourse by suggesting actionable changes on how to change the input to obtain a different (i.e. more favorable) system output. However, recent work highlighted their vulnerability to different types of manipulations.

This work studies the vulnerability of counterfactual explanations to data poisoning. We formally introduce and investigate data poisoning in the context of counterfactual explanations for increasing the cost of recourse on three different levels: locally for a single instance, or a sub-group of instances, or globally for all instances. In this context, we characterize and prove the correctness of several different data poisonings. We also empirically demonstrate that state-of-the-art counterfactual generation methods and toolboxes are vulnerable to such data poisoning.

1 Introduction

Many Artificial Intelligence (AI-) and Machine Learning (ML-) based systems are increasingly deployed in the real world (49; 20). These systems show an impressive performance but are still not perfect – e.g. failures, issues of fairness, and vulnerability to data poisoning can cause harm when applied in the real world. Given the threat of failures (intentionally caused or not), transparency of such deployed AI- and ML-based systems becomes a crucial aspect. Transparency is important not only to prevent failures but also to create trust in such systems and understand where and how it is safe to deploy them. The importance of transparency was also recognized by the policymakers and therefore found its way into legal regulations such as the EU's GDPR (14) or the more recent EU AI act (13). Explanations are a popular way of achieving transparency and shaping the field of eXplainable AI (XAI) (16). However, because of many different use cases and users, many different explanation methods exist (16; 2; 1; 33). One popular type of explanation method is counterfactual explanations (45), which are inspired by human explanations (11) and can be used to provide recourse to individuals. A counterfactual explanation provides (computational) recourse by stating actionable recommendations on how to change the system's output in some desired way – e.g. how to change a rejected loan application into an accepted one. Recent works have shown that counterfactual explanations are neither robust to model changes (28), nor to input perturbations (4; 43), and also

Preprint. Under review.



(a) Illustrative example of loan application: Poisoned training data set leads to a higher cost of recourse.

(b) Data poisoning at three different levels.

Figure 1

not to adversarial training (38). However, the vulnerability of counterfactual explanation methods to data poisoning remains unexplored. Data poisoning affects models in the training stage by changing training samples or adding new instances such that, for instance, the performance (e.g. accuracy) of the final trained model decreases (24; 41), or fairness issues arise (27; 39). Data poisoning can be done offline (24) or online (41). It only makes small changes to the training data such as changing labels, removing samples, or adding new instances, that are likely to remain unnoticed. This poses a real threat in practice because nowadays many large models are trained on huge data sets – often based on public data from the internet (49; 20) – where it is impossible to check data in detail and therefore poisonous data might affect a large number of models trained directly on the data or indirectly using some pre-trained embeddings or models (35; 8; 47). In the context of counterfactual explanations, data poisoning could increase the cost of recourse as illustrated in Figure 1a – either globally for all individuals or for a subset of individuals. Since counterfactual explanations state actionable recommendations that are to be executed in the real-world, manipulated explanations would directly affect the individuals by enforcing more costly actions or hiding some information from them. Although counterfactual explanations are a popular and widely used explanation method, the effect of data poisoning on them has not been studied yet.

In this work, we study the vulnerability of counterfactual explanations to data poisoning. We formalize and identify a set of data poisoning mechanisms for counterfactual explanations which injects a small set of realistic but poisonous data points into the training data set such that the decision boundary of a newly trained classifier changes to increase the cost of recourse on one of three different levels: locally for an individual or, a sub-group of individuals, or globally for all individuals. The method only needs access to an interface for getting predictions and a mechanism for generating closest counterfactuals, but no access or knowledge about model internals is needed. We empirically find that existing state-of-the-art methods for computing counterfactuals are vulnerable to data poisoning.

Related Work Most existing work (6) on exposing the vulnerability of explanations is centered in the vision domain and focuses either on adversarial examples or model manipulation. Only very little work considers domain-independent data poisoning (6). For instance, there exist data poisoning against partial dependence plots (7), SHAP (5), and concept-based explainability tools (10). The authors of (7) propose a genetic algorithm for perturbing the training data such that SHAP importances or attributions change. Their proposed method assumes that it is possible to change (possibly) all samples in the training set, which might constitute a very strong and unrealistic assumption in reality. Furthermore, changing many (or all) samples in the training data set might harm the model's predictive performance – this, however, is not evaluated in (7). A similar approach, with the same limitations, is proposed in (5) where partial dependence plots are targeted. In the context of counterfactual explanations, the authors of (38) propose an adversarial training objective such that the cost of recourse decreases for a sub-group of individuals. Note that this approach is model-specific and different from data poisoning since it proposes the use of a malicious cost function and therefore assumes full control over the training procedure. In this work, we consider data poisonings and argue that changing or adding training instances might often be more actionable in practice.

2 Foundations of Counterfactual Explanations and Recourse

A counterfactual explanation (often just called counterfactual) states actionable changes to the features of a given instance such that the system's output changes. Usually, an explanation is requested in the case of an unexpected or unfavorable outcome (34) – in the latter case, a counterfactual is also referred to as *recourse* (21), i.e. recommendations on how to change the unfavorable into a favorable outcome. Because counterfactuals can mimic ways in which humans explain (11), they constitute among one of the most popular explanation methods in literature and in practice (29; 42). There are two important properties that must be considered when formalizing and computing counterfactual explanations (45): 1) the contrasting property, requiring that the stated changes indeed change the output of the system; and 2) the cost of the counterfactual – i.e. the cost and effort it takes to execute the counterfactual in the real world should be as low as possible in order to maximize its usefulness (e.g. counterfactuals with very few changes or as small as possible changes). Both properties can be combined into an optimization problem (see Definition 1).

Definition 1 ((Closest) Counterfactual Explanation)**.** *Assume a classifier (binary or multi-class) $h : \mathbb{R}^d \to \mathcal{Y}$ is given. Computing a counterfactual $\vec{\delta}_{cf} \in \mathbb{R}^d$ for a given instance $\vec{x}_{orig} \in \mathbb{R}^d$ is phrased as the following optimization problem:* $\underset{\vec{\delta}_{cf} \in \mathbb{R}^d}{\arg\min}\ \ell\left(h(\vec{x}_{orig} + \vec{\delta}_{cf}), y_{cf}\right) + C \cdot \theta(\vec{\delta}_{cf})$ *where $\ell(\cdot)$ penalizes deviation of the prediction $h(\vec{x}_{cf} := \vec{x}_{orig} + \vec{\delta}_{cf})$ from the requested outcome y_{cf}, $\theta(\cdot)$ states the cost of the explanation (e.g. cost of recourse) which should be minimized, and $C > 0$ denotes the regularization strength balancing the two properties. CF$(\vec{x}, h)$ denotes the counterfactual $\vec{\delta}_{cf}$ of an instance $\vec{x}$ under a classifier $h(\cdot)$ iff the target outcome y_{cf} is uniquely determined.*

Note that the cost of the counterfactual, here modeled by $\theta(\cdot)$, is highly domain and use-case specific and therefore must be chosen carefully in practice and might require domain knowledge. In many implementations and toolboxes (19), the p-norm is used as a default.

Remark 1. *In the case of recourse – i.e. a counterfactual explanation $\vec{\delta}_{cf}$ (Definition 1) for turning an unfavorable into a favorable outcome –, we refer to the cost $\theta(\vec{\delta}_{cf})$, as the cost of recourse.*

In this work, w.l.o.g., we refer to $y = 0$ as the unfavorable, and $y = 1$ as the favorable outcome. Besides those two essential properties (contrasting and cost), there exist additional relevant aspects such as plausibility (26; 31), diversity (30), robustness (46; 48; 23; 4), fairness (3; 44; 37; 36), etc. which have been addressed in literature (19). There exist numerous methods and implementations/toolboxes for computing counterfactual explanations in practice (19) – most include some additional aspects such as plausibility, diversity, etc.: *FACE* (31) is a model-agnostic algorithm for computing feasible and actionable counterfactuals. Instead of computing a single counterfactual only, the method also outputs a path of intermediate actionable steps that lead from the original instance to the final counterfactual. *Counterfactuals Guided by Prototypes* (26) is another method focusing on plausibility. Here a set of plausible instances (prototypes) are used to pull the final counterfactual instance (i.e. $\vec{x}_{\text{orig}} + \vec{\delta}_{\text{cf}}$) closer to these plausible instances and by this make them more plausible. *DiCE* (30) is a model-agnostic method for computing a set of diverse closest counterfactual explanations instead of a single one only. *Nearest Training Sample method* constitutes a straightforward baseline method for computing plausible counterfactual explanations that can be implemented by picking the closest sample, with the requested output y_{cf}, from a given set (e.g. training set) as the counterfactual instance.

3 Data Poisoning of Counterfactual Explanations

Data poisoning of counterfactuals can have effects on different levels: all individuals are affected (global effect), only some sub-groups are affected (sub-groups effect), or only a single individual is affected (local effect). Also, data poisoning can aim for different effects on counterfactuals , such as hiding attributes or increasing the cost of recourse (Remark 1). Since providing (computational) recourse is a core application of counterfactuals, increasing the cost of recourse has the most severe consequence in the real world because it would harm individuals directly by making the recourse more costly. Therefore, in this work, we focus on data poisoning for increasing the cost of recourse.

3.1 Data Poisoning for Increasing the Cost of Recourse

In this work, we study the effect of data poisoning on the cost of recourse (Remark 1). That is, we focus on data poisoning with the primary goal of increasing the cost of recourse, in a pre-defined region in data space, as stated in Definition 2.

Definition 2 (Data Poisoning for Increasing the Cost of Recourse)**.** *Given an original training data set $\mathcal{D}_{orig} \subset \{\mathcal{X} \times \mathcal{Y}\}^n$ and a probability density $\phi(\cdot)$ assigning a high likelihood to targeted instances, we transform (i.e. poison) $\mathcal{D}_{orig}$ into a new data set $\mathcal{D}_{poisoned} \subset \{\mathcal{X} \times \mathcal{Y}\}^m$ by means of a data poisoning mechanism $T : \{\mathcal{X} \times \mathcal{Y}\}^n \to \{\mathcal{X} \times \mathcal{Y}\}^m$, such that the cost of recourse $\theta(\cdot)$ increases for instances under $\phi(\cdot)$:*

$$\mathbb{E}_{\vec{x}\sim\phi}\left[\theta \circ CF(\vec{x}, h_{\mathcal{D}_{poisoned}})\right] > \mathbb{E}_{\vec{x}\sim\phi}\left[\theta \circ CF(\vec{x}, h_{\mathcal{D}_{orig}})\right] \text{ where } \mathcal{D}_{poisoned} = T(\mathcal{D}_{orig}) \tag{1}$$

where $\circ$ denotes the function composition, $h_{\mathcal{D}}$ denotes a classifier that was derived from the data set $\mathcal{D}$, and CF$(\cdot,\cdot)$ a method for generating counterfactuals.

The density $\phi(\cdot)$ allows us to vary the level of the poisoning – e.g. for a global effect, we could use a class-wise density for targeting all instances from a specific class, or in the case of a local effect, we could use a delta-density to target a single instance or a small group of instances. In this work, we focus on data poisoning attacks $T(\cdot)$ that add new (poisonous) instances to the training data set to increase the cost of recourse (Definition 2). In this context, we formally define (Definition 3) a poisonous data set (40) for increasing the cost of recourse – i.e. a data set that increases the cost of recourse (Definition 2) if added to the original training data $\mathcal{D}_{\text{orig}}$.

Definition 3 (Poisonous Data Set)**.** *We say that a data set $\mathcal{D}_{poison} \subset \{\mathcal{X} \times \mathcal{Y}\}^m$ is recourse poisoning for a training data set $\mathcal{D}_{orig} \subset \{\mathcal{X} \times \mathcal{Y}\}^n$ iff a classifier $h : \mathcal{X} \to \mathcal{Y}$ trained on $\mathcal{D}_{poison} \cup \mathcal{D}_{orig}$ shows an increase in the cost of recourse (Definition 2):* $\mathbb{E}_{\vec{x}\sim\phi}\left[\theta \circ CF(\vec{x}, h_{\mathcal{D}_{poison}\cup\mathcal{D}_{orig}})\right] > \mathbb{E}_{\vec{x}\sim\phi}\left[\theta \circ CF(\vec{x}, h_{\mathcal{D}_{orig}})\right]$

Consequently, we write the data poisoning mechanisms $T(\cdot)$ (Definition 2) as follows: $T(\mathcal{D}_{\text{orig}}) = \mathcal{D}_{\text{orig}} \cup \mathcal{D}_{\text{poison}}$ where $\mathcal{D}_{\text{poison}}$ refers to a poisonous data set from Definition 3. From a practical point of view, besides increasing the cost of recourse (as stated in Definition 2), the poisonous data set $\mathcal{D}_{\text{poison}}$ (Definition 3) should have the following properties:

1. The number of poisonous instances $\mathcal{D}_{\text{poison}}$ is small: $\arg\min\ |\mathcal{D}_{\text{poison}}|$
2. The poisonous instances $\mathcal{D}_{\text{poison}}$ are realistic – i.e. they are on the data manifold $p_{\text{data}}(\cdot)$ and have a high likelihood: $\arg\max\ p_{\text{data}}(\vec{x}_i, y_i) \quad \forall\,(\vec{x}_i, y_i) \in \mathcal{D}_{\text{poison}}$
3. In the case of aiming for a local or sub-group effect, poisonous instances only target those specified groups, but do not affect any other instances – i.e. the cost of recourse of untargeted instances should not change: $\mathbb{E}_{\vec{x}\sim\phi'}\left[\theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{poison}}\cup\mathcal{D}_{\text{orig}}})\right] \approx \mathbb{E}_{\vec{x}\sim\phi'}\left[\theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{orig}}})\right]$ where ϕ' denotes the density of all untargeted instances.
4. The predictive performance of the classifier is maintained[1]: $\arg\min\ \mathbb{E}[\ell(h_{\mathcal{D}_{\text{poison}}\cup\mathcal{D}_{\text{orig}}}(\vec{x}_i)), y_i)]$ where $\ell(\cdot)$ denotes some suitable loss function.

Considering all these aspects, we formalize the finding of a poisonous data set $\mathcal{D}_{\text{poison}}$ (Definition 3) as a multi-objective optimization problem:

$$\underset{\mathcal{D}_{\text{poison}}}{\arg\min}\ \left(|\mathcal{D}_{\text{poison}}|, \mathbb{E}[\ell(h_{\mathcal{D}_{\text{poison}}}(\vec{x}_i)), y_i)], \sum_{(\vec{x}_i,y_i)\in\mathcal{D}_{\text{poison}}} -p_{\text{data}}(\vec{x}_i, y_i)\right) \tag{2a}$$

$$\text{s.t. } \mathbb{E}_{\vec{x}\sim\phi}\left[\theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{poison}}\cup\mathcal{D}_{\text{orig}}})\right] > \mathbb{E}_{\vec{x}\sim\phi}\left[\theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{orig}}})\right] \tag{2b}$$

$$\mathbb{E}_{\vec{x}\sim\phi'}\left[\theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{poison}}\cup\mathcal{D}_{\text{orig}}})\right] \approx \mathbb{E}_{\vec{x}\sim\phi'}\left[\theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{orig}}})\right] \tag{2c}$$

In the following, we study a few (general) aspects and properties of Eq. (2) and poisonous data sets (Definition 3) that will serve as a foundation for the final data poisoning algorithm (Algorithm 1).

Locally Increasing the Cost of Recourse As discussed in Section 2, the simplest way of achieving recourse is by means of the closest counterfactual as stated in Definition 1 – i.e. the smallest change

[1] However, because the decision boundary is changed, some drop in the predictive performance might be inevitable.

that reaches/crosses the decision boundary. Regarding local poisoning, it can be shown that, under some assumptions, samples on the decision boundary are data poisoning instances (Definition 3).

Theorem 1 (Local Recourse Poisoning Data Sets for 1-Nearest Neighbor Classifiers)**.** *Let $h_{\mathcal{D}}(\cdot)$ be a k-nearest neighbor classifier (with $k = 1$) for some data set $\mathcal{D}$. For any $(\vec{x}_{orig}, y_{orig}) \in \mathcal{D}$, let $\vec{x}'$ denote the closest instance (assuming uniqueness) on the decision boundary under a p-norm $\theta(\cdot)$. Then, $\mathcal{D}_{poison} = \{(\vec{x}', y_{orig})\}$ is a poisonous data set (Definition 3) at $\vec{x}_{orig}$ – i.e. the cost resources increases for $\vec{x}_{orig}$, i.e.: $\theta \circ CF(x, h_{\mathcal{D} \cup \{(\vec{x}', y_{orig})\}}) > \theta \circ CF(x, h_{\mathcal{D}})$*

Although a 1-NN classifier is somewhat simplistic, it is quite flexible and might be a good local approximation for many different classifiers. Therefore, Theorem 1 provides valuable insights on the nature of recourse poisoning data sets (Definition 3) for locally increasing the cost of recourse.

Globally Increasing the Cost of Recourse Similar to Theorem 1, it is possible, under some assumptions, to state a poisonous data set (Definition 3) in the case of a linear Support Vector Machine classifier (SVM) for globally increasing the cost of recourse.

Theorem 2 (Global Recourse Poisoning Data Set for linear SVM)**.** *Let $h_{\mathcal{D}} : \mathbb{R}^d \to \{-1, 1\},\ \vec{x} \mapsto \text{sign}(\vec{w}^\top \vec{x} + b)$ be a linear SVM classifier, and assume that the training data set $\mathcal{D}$ is linearly separable. Then, a poisonous data set $\mathcal{D}_{poison}$ (Definition 3) for all negatively classified samples (i.e. $\forall\, \vec{x} \in \mathbb{R}^d :\ h(\vec{x}) = -1$) is given as follows: $\mathcal{D}_{poison} = \{(\vec{x}, -1) \mid \vec{x} \in \mathbb{R}^d$ with $-\vec{w}^\top \vec{x} - b \geq 1 - \xi,\ \xi \in (0, 1)\}$*

Note that Theorem 2 states that a recourse poisoning data set can be constructed by considering all samples inside the maximum margin of $h(\cdot)$. In practice, however, it is likely possible that already a small subset of $\mathcal{D}_{\text{poison}}$ constitutes a poisonous data set (Definition 3) as well.

3.2 Data Poisoning on Counterfactual Explanations

Based on the findings from Section 3.1, we formalize a method (see Algorithm 1) for generating poisonous data sets (Definition 3) – i.e. poisonous instances that are added to the training set, to increase the cost of recourse. Consequently, our proposed Algorithm 1 constitutes and implementation of a data poisoning $T(\cdot)$ from Definition 2. Note that this proposed method supports data poisonings on different levels (i.e. local, sub-groups, and global levels).

For practical purposes, we assume that we have (or created) a set of samples $\mathcal{D}_{\text{target}} = \{(\vec{x}_j, y)\}$ all with the same prediction $y \in \{0, 1\}$, where $\vec{x}_j \sim \phi$, from the region in data space that is targeted by the poisoning – e.g. this could be a subset of the training data set. We propose to fix the size of the poisonous data set (Definition 3) (i.e. the number of poisonous instances $\{\vec{z}_i\}$ is fixed) and approximate the original multi-objective optimization problem Eq. (2) by Eq. (3).

$$\underset{\{\vec{z}_i\}}{\arg\min} \left(\underset{\vec{x}_j \in \mathcal{D}_{\text{target}}}{\arg\min} \|\vec{z}_i - \vec{x}_j\|_p, \mathbb{E}[\ell(h_{\mathcal{D}_{\text{orig}} \cup \mathcal{D}_{\text{poison}}}(\vec{x}_l)), y_l)] \right) \tag{3a}$$

$$\text{s.t.} \sum_{\vec{x} \in \mathcal{D}_{\text{target}}} \theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{orig}} \cup \mathcal{D}_{\text{poison}}}) > \sum_{\vec{x} \in \mathcal{D}_{\text{target}}} \theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{orig}}}) \tag{3b}$$

$$\mathbb{E}_{\vec{x} \sim \phi'} \left[\theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{orig}} \cup \mathcal{D}_{\text{poison}}}) \right] \approx \mathbb{E}_{\vec{x} \sim \phi'} \left[\theta \circ \text{CF}(\vec{x}, h_{\mathcal{D}_{\text{orig}}}) \right] \tag{3c}$$

where the poisonous data set (Definition 3) $\mathcal{D}_{\text{poison}}$ is constructed as $\mathcal{D}_{\text{poison}} = \{(\vec{z}_i, y)\}$.

Note that the objective in Eq. (3) replaces the original plausibility constraint in Eq. (2). That is, we construct poisonous instances that are very similar to the given samples $\mathcal{D}_{\text{target}}$ – note that it was observed (12) that small perturbations often remain unnoticed by the human, which gave rise to adversarial attacks (12; 32). By this, we aim to make the poisonous instances more difficult to detect.

Implementation We propose to compute an approximate solution to Eq. (3) by constructing instances $\vec{z}_i$ that are on the decision boundary or behind it and are close to samples in $\mathcal{D}_{\text{target}}$ – Theorem 1 and Theorem 2 suggest that those samples form a poisonous data set (Definition 3). We construct such instances by computing closest counterfactual explanations δ_i (Definition 1) of samples $(\vec{x}_j, y) \in \mathcal{D}_{\text{target}}$ that are close to the decision boundary:

$$\vec{z}_i = \vec{x}_j + \delta_j \quad \text{for some } \vec{x}_j \text{ where } \delta_j \text{ is "small"} \tag{4}$$

where we estimate the distance δ_i to the decision boundary by computing a closest counterfactual – i.e. $\delta_i = \text{CF}(\vec{x}_i, h)$. Note that the counterfactual $\vec{\delta}_{\text{cf}}$ used in Eq. (4) is *not* necessarily computed

Algorithm 1 Data Poisoning for Increasing the Cost of Recourse

Input: Samples $\mathcal{D}_{\text{target}} = \{(\vec{x}_i, y)\}$ from the data space region that is targeted; Mechanism $\text{CF}(\cdot, h)$ for generating closest counterfactuals; Number n of poisonous instances; Parameters: k, b
Output: Poisoned training data set D_{poisoned}

1: $\{\delta_i = \theta \circ \text{CF}(\vec{x}_i, h) \quad \forall \vec{x}_i \in \mathcal{D}_{\text{target}}\}$ ▷ Estimate distances to decision boundary
2: $D_{\text{poison}} = \{\}$
3: **for** n-times **do**
4: $(\vec{x}, y) \sim$ weighted_sampling $(\mathcal{D}_{\text{target}}, \{\delta_i\})$ ▷ Prefer samples close to the decision boundary
5: $\Delta_{\text{cf}} = \text{CF}(\vec{x}, h; k)$ ▷ k diverse closest CFs
6: **for** $\vec{\delta}_{\text{cf}} \in \Delta_{\text{cf}}$ **do**
7: **for** $\alpha \in [1, b]$ **do**
8: $\vec{z} = \vec{x} + \alpha * \vec{\delta}_{\text{cf}}$ ▷ Add samples along $\vec{\delta}_{\text{cf}}$
9: $D_{\text{poison}} = D_{\text{poison}} \cup \left\{ (\vec{z}, y) \right\}$
10: **end for**
11: **end for**
12: **end for**
13: $D_{\text{poisoned}} = D_{\text{train}} \cup D_{\text{poison}}$ ▷ Add D_{poison} to training set

by the same counterfactual generation method that is targeted by the data poisoning (Definition 3). Maintaining predictive performance objective in Eq. (2) and not changing the cost of recourse should for untargeted instance, are both considered implicitly in Eq. (4) – because the poisonous instances $\vec{z}_i$ are close to the targeted instances in $\mathcal{D}_{\text{target}}$, a sufficiently flexible classifier should not change its behavior in other regions in data space. Furthermore, because we only consider samples $\vec{x}_j$ that are close to the decision boundary, the corresponding $\vec{z}_i$ (which constitute a counterfactual instance of $\vec{x}_j$) are expected to be very similar to $\vec{x}_j$ and therefore satisfying the plausibility constraint in Eq. (2). To increase the robustness of the poisoning, we propose to use not only a single closest counterfactual in Eq. (4) but a set of diverse closest counterfactual explanations. We also propose to extend the counterfactual direction $\vec{\delta}_{\text{cf}}$ by multiplying it with a factor $\alpha > 1$, to create an even larger increase in the cost of recourse. The pseudo-code of data poisoning is given in Algorithm 1.

Correctness From Theorem 1 it follows (see Corollary 1) that Algorithm 1 computes valid poisonous data sets for a k-NN classifier with $k = 1$.

Corollary 1 (Correctness of Algorithm 1)**.** *Let $h_{\mathcal{D}}(\cdot)$ be a k-nearest neighbor classifier (with $k = 1$) for some data set $\mathcal{D}$. For any $\mathcal{D}_{target} = \{(\vec{x}_{orig}, y_{orig})\}$ where $(\vec{x}_{orig}, y_{orig}) \in \mathcal{D}$, Algorithm 1 computes a poisonous data set (Definition 3) that increases the cost of recourse of $\vec{x}_{orig}$.*

Although k-NN classifiers with $k = 1$ might seem simplistic, they constitute a flexible classifier that might be used as an approximation of other more sophisticated classifiers.

Runtime The runtime of Algorithm 1 can be broken down to $\mathcal{O}(n \cdot k \cdot \rho)$ where n and k are the hyper-parameters of the algorithm referring to the number of poisonous instances (i.e. size of the poisonous data set), and ρ denotes the computational complexity (i.e. runtime) for computing a counterfactual which is utilized to construct the poisonous sample(s) Eq. (4) (see line 8 in Algorithm 1) – note ρ is likely to differ between different counterfactual generation mechanisms. Consequently, the runtime of Algorithm 1 scales linearly with the number of requested poisonous samples.

Limitations The major limitation of Algorithm 1 is that it requires access to a counterfactual generation method for generating poisonous instances. A gradient-based approximation of the counterfactual generation method could be an alternative to this. However, besides assuming gradients (not possible for tree-based models), one would lose the correctness guarantees. Furthermore, our approach still constitutes a more realistic assumption compared to (38) where an attacker is assumed to be able to manipulate the loss function used in training.

4 Experiments

We empirically evaluate the robustness of counterfactual explanations (i.e. recourse) against data poisonings by applying the data poisoning Algorithm 1 from Section 3 on combinations of several

Table 1: Difference in the cost of recourse: no poisoning vs. *global poisoning*. The amount of poisoning (percentage %) is specified for each method separately. Positive numbers denote an increase in the cost of recourse, while negative numbers denote the opposite. We report the median (over all folds) rounded to two decimal places.

Classifier	Data set	Nearest $^{10\%}$	DiCE $^{10\%}$	FACE $^{40\%}$	Proto $^{20\%}$
SVC	Credit	4.94	7.59	−0.91	8.33
	Diabetes	2.28	2.23	−0.03	2.46
	Crime	9.81	9.45	15.24	14.22
RNF	Credit	3.56	4.9	2.35	6.05
	Diabetes	0.64	0.91	−0.24	1.84
	Crime	4.32	6.85	11.17	13.65
DNN	Credit	1.06	1.93	2.36	0.38
	Diabetes	0.68	0.66	1.6	1.24
	Crime	5.18	6.65	11.81	9.04

different benchmark data sets, classifiers, and state-of-the-art counterfactual explanation generation methods and toolboxes. The Python implementation of the experiments (including all data sets), is available on GitHub[2].

4.1 Benchmark Data Sets & Machine Learning Classifiers

We consider three data sets that all contain a sensitive attribute such that we can also evaluate the difference in the cost of recourse between protected groups: The "Diabetes" data set (17) (denoted as *Diabetes*) contains data from 442 diabetes patients, each described by 9 numeric attributes and the sensitive attribute "sex" of each patient. The target is a binarized quantitative measure of disease progression one year after baseline. The "Communities & Crime" data set (15) (denoted as *Crime*) contains 1994 socio-economic data, including the sensitive attribute "race", records from the USA. Following the pre-processing in (22), we are left with 100 encoded attributes to predict the crime rate (low vs. high). The "German Credit Data set" (18) (denoted as *Credit*) is a data set for loan approval and contains 1000 instances each annotated with 7 numerical and 13 categorical attributes, including the sensitive attribute "sex", with a binary target value. We use only the seven numerical features.

We consider a diverse set of ML classifiers $h(\cdot)$: deep neural networks (denotes as *DNN*), random forests (denoted as *RNF*), and linear SVM 's(denoted as *SVC*).

4.2 Counterfactual Generation Methods

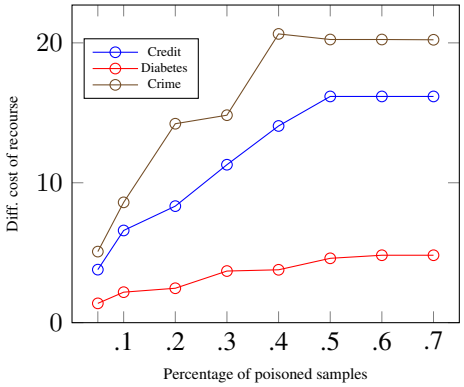
Given the large amount of different counterfactual generation methods (19), we evaluate the data poisoning method on a set of very different and popular state-of-the-art methods for computing computational recourse: Nearest Training Sample (denoted as *Nearest*), as simple baseline; FACE (31) and Counterfactuals guided by Prototypes (26) (denoted as *Proto*) for computing plausible counterfactuals; DiCE (30) for diverse closest counterfactuals.

4.3 Setup

In all experiments (as described below), we use DiCE (30) as a counterfactual generation mechanism for computing three diverse closest counterfactuals (i.e. $k = 3$ in Algorithm 1), that are as close as possible to the original sample. We use the ℓ_1 norm as a popular implementation (19) of the cost of recourse – i.e. $\theta(\cdot) = \|\cdot\|_1$. Furthermore, all experiments are run in 5-fold cross-validation. In all global and sub-group data poisoning scenarios, we evaluate different amounts (5% to 70%) of poisonous instances – i.e. original training data + poisoned instances. We not only evaluate the influence of the number of poisonous instances on the cost of recourse, but also their influence on the classifiers' predictive performance – some of these results are shown in Figures 2,3a while the rest can be found in the appendix. Note that, although we use the DiCE method (30) for constructing the poisonous instances in Algorithm 1, it can still be considered a model-agnostic method because it is also able to attack other counterfactual generation methods.

[2] `https://github.com/andreArtelt/DataPoisoningCounterfactuals`

(a) $h(\cdot)$: SVC - CF: *Proto*



(b) $h(\cdot)$: SVC - CF: *Proto*

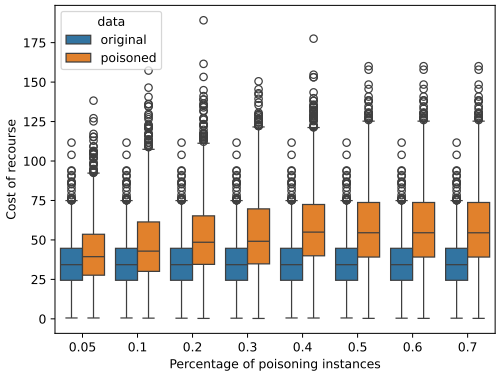


Figure 2: Global data poisoning: a) Median (over all folds) difference in the cost of recourse vs. percentage of poisoned instances. b) Cost of recourse vs. percentage of poisoned instances.

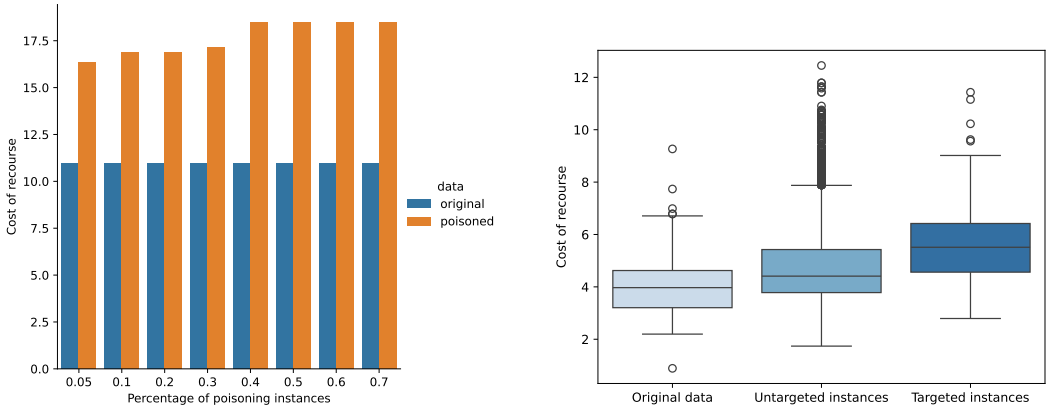
Table 2: Difference in the cost of recourse between protected groups: no vs. poisoning on a *sub-group level*. The amount of poisoning (percentage %) is specified for each method separately. Positive numbers denote an increase in the difference in the cost of recourse, while negative numbers denote the opposite. We report the median (over all folds) rounded to two decimal places.

Classifier	Data set	Nearest $^{10\%}$	DiCE $^{40\%}$	FACE $^{40\%}$	Proto $^{50\%}$
SVC	Credit	0.78	0.08	0.73	9.79
	Diabetes	1.23	0.54	0.02	2.85
	Crime	5.7	7.51	0.46	12.4
RNF	Credit	0.48	0.12	3.18	0.07
	Diabetes	0.72	1.59	0.05	1.09
	Crime	4.62	6.96	14.32	1.27
DNN	Credit	0.13	0.47	0.22	9.11
	Diabetes	1.1	0.75	−0.2	2.29
	Crime	9.06	10.09	1.86	0.47

Data poisoning on a global level For every, negative classified, sample in the test set, we compute a counterfactual explanation. We evaluate the global increase in the cost of recourse, by computing the difference in the cost of recourse: $\theta \circ \text{CF}(\vec{x}_i, h_{\mathcal{D}_{\text{orig}} \cup \mathcal{D}_{\text{poison}}}) - \theta \circ \text{CF}(\vec{x}_i, h_{\mathcal{D}_{\text{orig}}})$. A positive score means an increase in the recourse cost (due to the data poisoning), while a negative or near zero score implies no change or a lower cost of recourse. We report the median to avoid the influence of outliers. In Table 1 we show the increase in the cost of recourse together with the amount of poisoning that was necessary for observing a significant increase – more detailed results are provided in the appendix.

Data poisoning on a sub-group level We consider sub-groups created based on the sensitive attribute – note that this is a reasonable but only one out of many possible ways how sub-groups might be created. We apply the data poisoning to poison instances from one protected group only, assuming that the sensitive attribute of each instance is known. By this, we aim to increase the difference in the cost of recourse between the two protected groups – i.e. group-unfairness in recourse (3; 44; 37).

For every, negative classified, sample in the test set (no matter to which sub-group it belongs), we compute a counterfactual. We evaluate the difference in the cost of recourse between the two sub-groups as follows: $s_{\mathcal{D}_{\text{orig}} \cup \mathcal{D}_{\text{poison}}} - s_{\mathcal{D}_{\text{orig}}}$ with $s_{\mathcal{D}} = \|\theta \circ \text{CF}(\vec{x}_i | s = 0, h_{\mathcal{D}}) - \theta \circ \text{CF}(\vec{x}_i | s = 1, h_{\mathcal{D}})\| \quad \forall\, \vec{x}_i \in \mathcal{D}_{\text{test}}\ h(\vec{x}_i) = 0$ where we denote the sensitive attribute as s – i.e. $\vec{x}_i | s = 0$ means that we only consider x_i if its sensitive attribute is equal to zero. A positive score refers to an increase in the difference of the cost of recourse between the protected groups, while a negative score refers to the opposite. Furthermore, note that we use the median (over all folds) to avoid the influence of outliers. We show the results together with the minimum amount of poisoning that was necessary for observing a significant increase in Table 2 – more detailed results are provided in the appendix.



(a) $h(\cdot)$: SVC - CF: *DiCE* (b) CF: *Nearest*

Figure 3: a) Sub-group data poisoning on the *Crime* data set – no poisoning vs. different percentages of poisoned instances. Difference in the median cost of recourse between the two protected groups. b) *Local* data poisoning on the diabetes data set and a DNN classifier.

Data poisoning on a local level We compute a local data poisoning for every, negative classified, sample in the test set. However, because of computational limitations – i.e. for every sample in the test set (over all folds), the entire data poisoning has to be run and evaluated –, we only evaluate a single scenario considering a DNN classifier on the diabetes data set. Some of the results are shown in Figure 3b – more detailed results are given in the appendix.

Detection of poisonous instances To hinder their detection, the poisonous instances (Definition 3) should look like plausible training samples. We evaluate this by applying two classic outlier detection methods to the poisoned training data set $\mathcal{D}_{\text{orig}} \cup \mathcal{D}_{\text{poison}}$: Isolation Forest (25) and the Local Outlier Factor method (9). Both methods are calibrated on the (unpoisoned) test set and applied to $\mathcal{D}_{\text{orig}} \cup \mathcal{D}_{\text{poison}}$ to detect the poisonous instances $\mathcal{D}_{\text{poison}}$. We compare the performance of those two detection methods to the baseline of predicting a poisonous sample with the ground-truth rate of poisonous samples. The detailed results (precision and recall) are given in the appendix.

4.4 Results & Discussion

General trend We observe that in almost all scenarios, on local as well as on global levels (see Tables 1,2 and appendix), even a relatively small amount of poisonous instances, added to the training data set, leads to a significant increase in the cost of recourse. Increasing the number of poisonous instances leads to an even larger increase in the cost of recourse (see Figure 2 and appendix). However, we observe differences in the necessary amount of poisonous instances between different counterfactual generation methods and toolboxes. For FACE (31) and counterfactuals guided by prototypes (26), we need significantly more poisonous instances for increasing the cost of recourse – in the case of FACE, we even have a few settings (in particular for SVC) where the poisoning does not work which is likely due to the special nature of FACE that might require a different strategy. Since both methods focus on plausibility, this might be an indicator that additional plausibility constraints can act as a beneficial regularize for increased stability. Altogether, the results demonstrate the vulnerability of existing (state-of-the-art) counterfactual generation methods to data poisonings.

Data poisoning on a sub-group level In the case of sub-groups, we observe (see Table 2) a similar effect compared to the global and local poisoning. However, the increases are not as large as those for the local or global poisoning and often the necessary amount poisonous instances is also larger compared to the global poisoning – this is quite likely due to a strong overlap of the distributions of the sub-groups which makes it difficult to just change the cost of recourse for one group but not for the other. Furthermore, it is worth noting that in many cases the initial difference in the cost of recourse is already quite significant (see Figure 3a and appendix).

Effect on the predictive performance We observe the expected results that classifiers' predictive performance is decreasing the more poisoned instances are added – i.e. for a global data poisoning the decrease in predictive performance is worse than for sub-group or local data poisonings.

Detection of poisonous instances Concerning the detectability of the generated poisonous instances, we observe that both outlier detection methods are not able to reliably detect the poisonous instances. Furthermore, while the detection precision is often not too bad, for the recall both methods are outperformed by the baseline method of randomly predicting poisonous instances. These findings demonstrate that the detection of our generated poisonous instances is non-trivial.

5 Conclusion & Summary

In this work, we studied the robustness of counterfactuals against data poisonings. We identified and formalized data poisonings to increase the cost of recourse on different levels (local - global) by adding poisonous instances to the training data. We observed that in almost all cases the injection of already a small portion of poisonous instances into the training data leads to a significant increase in the cost of recourse on all levels. These findings demonstrate how easily existing classifiers and state-of-the-art counterfactual generation methods can be fooled by manipulating the training data. Since counterfactuals are a widely used method for providing recourse and analyzing ML-based models, (malicious) manipulations of those directly harm the user and consequently significantly reduce users' trust in this XAI method. Thus, this work demonstrates the necessity of more robust counterfactual generation methods as well as defense mechanisms against malicious data manipulations – we leave this as future research together with the exploration of other data poisoning mechanisms.

Acknowledgments and Disclosure of Funding

This research was supported by the Ministry of Culture and Science NRW (Germany) as part of the Lamarr Fellow Network. This publication reflects the views of the authors only.

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

References

[1] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.

[2] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

[3] André Artelt and Barbara Hammer. "explain it in the same way!" – model-agnostic group fairness of counterfactual explanations. In Amir Ofra, Tim Miller, and Hendrik Baier, editors, *Workshop on XAI*, 2023.

[4] André Artelt, Valerie Vaquet, Riza Velioglu, Fabian Hinder, Johannes Brinkrolf, Malte Schilling, and Barbara Hammer. Evaluating robustness of counterfactual explanations. In *2021 IEEE Symposium Series on Computational Intelligence*, pages 01–09. IEEE, 2021.

[5] Hubert Baniecki and Przemyslaw Biecek. Manipulating shap via adversarial data perturbations (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 12907–12908, 2022.

[6] Hubert Baniecki and Przemyslaw Biecek. Adversarial attacks and defenses in explainable artificial intelligence: A survey. *Information Fusion*, page 102303, 2024.

[7] Hubert Baniecki, Wojciech Kretowicz, and Przemyslaw Biecek. Fooling partial dependence via data poisoning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 121–136. Springer, 2022.

[8] Aleksandar Bojchevski and Stephan Günnemann. Adversarial attacks on node embeddings via graph poisoning. In *International Conference on Machine Learning*, pages 695–704. PMLR, 2019.

[9] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[10] Davis Brown and Henry Kvinge. Making corgis important for honeycomb classification: Adversarial attacks on concept-based explainability tools. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 620–627, 2023.

[11] Ruth M. J. Byrne. Counterfactuals in explainable artificial intelligence (xai): Evidence from human reasoning. In *IJCAI-19*, 2019.

[12] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. A survey on adversarial attacks and defences. *CAAI Transactions on Intelligence Technology*, 6(1):25–45, 2021.

[13] European Commission, Directorate-General for Communications Networks, Content, and Technology. Proposal for a Regulation laying down harmonised rules on Artificial Intelligence (Artificial Intelligence Act) and amending certain Union legislative acts. *Policy and Legistlation*, 21-04-2021.

[14] Council of European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC. *Official Journal of the European Union*, L 119:4.5, 2016.

[15] Dua Dheeru and E Karra Taniskidou. Uci machine learning repository. 2017.

[16] Rudresh Dwivedi, Devam Dave, Het Naik, Smiti Singhal, Rana Omer, Pankesh Patel, Bin Qian, Zhenyu Wen, Tejal Shah, Graham Morgan, et al. Explainable ai (xai): Core ideas, techniques, and solutions. *ACM Computing Surveys*, 55(9):1–33, 2023.

[17] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. 2004.

[18] Statlog (german credit data) data set, 1994.

[19] Riccardo Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, pages 1–55, 2022.

[20] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *The Journal of Machine Learning Research*, 23(1):2249–2281, 2022.

[21] Amir-Hossein Karimi, Gilles Barthe, Bernhard Schölkopf, and Isabel Valera. A survey of algorithmic recourse: contrastive explanations and consequential recommendations. *ACM Computing Surveys*, 2021.

[22] Tai Le Quy, Arjun Roy, Vasileios Iosifidis, Wenbin Zhang, and Eirini Ntoutsi. A survey on datasets for fairness-aware machine learning. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, page e1452, 2022.

[23] Francesco Leofante and Nico Potyka. Promoting counterfactual robustness through diversity. *arXiv preprint arXiv:2312.06564*, 2023.

[24] Jing Lin, Long Dang, Mohamed Rahouti, and Kaiqi Xiong. Ml attack models: adversarial attacks and data poisoning attacks. *arXiv preprint arXiv:2112.02797*, 2021.

[25] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[26] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. pages 650–665, 2021.

[27] Ninareh Mehrabi, Muhammad Naveed, Fred Morstatter, and Aram Galstyan. Exacerbating algorithmic bias through fairness attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8930–8938, 2021.

[28] Saumitra Mishra, Sanghamitra Dutta, Jason Long, and Daniele Magazzeni. A survey on the robustness of feature importance and counterfactual explanations. *arXiv preprint arXiv:2111.00358*, 2021.

[29] Christoph Molnar. *Interpretable Machine Learning*. 2019.

[30] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 607–617, 2020.

[31] Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. Face: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350, 2020.

[32] Jonas Rauber, Roland Zimmermann, Matthias Bethge, and Wieland Brendel. Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax. *Journal of Open Source Software*, 5(53):2607, 2020.

[33] Atul Rawal, James Mccoy, Danda B Rawat, Brian Sadler, and Robert Amant. Recent advances in trustworthy explainable artificial intelligence: Status, challenges and perspectives. *IEEE Transactions on Artificial Intelligence*, 1(01):1–1, 2021.

[34] Maria Riveiro and Serge Thill. The challenges of providing explanations of ai systems when they do not behave like users expect. In *Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization*, pages 110–120, 2022.

[35] Shawn Shan, Wenxin Ding, Josephine Passananti, Haitao Zheng, and Ben Y Zhao. Prompt-specific poisoning attacks on text-to-image generative models. *arXiv preprint arXiv:2310.13828*, 2023.

[36] Shubham Sharma, Jette Henderson, and Joydeep Ghosh. Certifai: A common framework to provide explanations and analyse the fairness and robustness of black-box models. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 166–172, 2020.

[37] Shubham Sharma, Alan H Gee, David Paydarfar, and Joydeep Ghosh. Fair-n: Fair and robust neural networks for structured data. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 946–955, 2021.

[38] Dylan Slack, Anna Hilgard, Himabindu Lakkaraju, and Sameer Singh. Counterfactual explanations can be manipulated. *Advances in Neural Information Processing Systems*, 34:62–75, 2021.

[39] David Solans, Battista Biggio, and Carlos Castillo. Poisoning attacks on algorithmic fairness. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 162–177. Springer, 2020.

[40] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. *Advances in neural information processing systems*, 30, 2017.

[41] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25*, pages 480–501. Springer, 2020.

[42] Sahil Verma, John Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review, 2020.

[43] Marco Virgolin and Saverio Fracaros. On the robustness of sparse counterfactual explanations to adverse perturbations. *Artificial Intelligence*, 316:103840, 2023.

[44] Julius Von Kügelgen, Amir-Hossein Karimi, Umang Bhatt, Isabel Valera, Adrian Weller, and Bernhard Schölkopf. On the fairness of causal algorithmic recourse. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9584–9594, 2022.

[45] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[46] Yongjie Wang, Hangwei Qian, Yongjie Liu, Wei Guo, and Chunyan Miao. Flexible and robust counterfactual explanations with minimal satisfiable perturbations. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 2596–2605, 2023.

[47] Ziqing Yang, Xinlei He, Zheng Li, Michael Backes, Mathias Humbert, Pascal Berrang, and Yang Zhang. Data poisoning attacks against multimodal encoders. In *International Conference on Machine Learning*, pages 39299–39313. PMLR, 2023.

[48] Songming Zhang, Xiaofeng Chen, Shiping Wen, and Zhongshan Li. Density-based reliable and robust explainer for counterfactual explanation. *Expert Systems with Applications*, 226:120214, 2023.

[49] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

A Appendix / supplemental material

B Proofs

B.1 Proof of Theorem 1

Proof. Sketch: For any $\vec{x}_{\text{orig}}, h(\vec{x}_{\text{orig}}) = y_{\text{orig}}$, assume uniqueness of the solution $\vec{x}'$ – i.e. the closest sample to $\vec{x}_{\text{orig}}$ on the decision boundary:

$$\begin{aligned} &\arg\min_{\vec{x}' \in \mathbb{R}^d} \|\vec{x}' - \vec{x}_{\text{orig}}\|_p \text{ s.t.} \\ &\exists i \neq j : (\vec{x}_i, y_i), (\vec{x}_j, y_j) \in \mathcal{D}, y_i \neq y_j, \text{ with } \|\vec{x}' - \vec{x}_i\|_p = \|\vec{x}' - \vec{x}_j\|_p \end{aligned} \tag{5}$$

where we (w.l.o.g.) assume the use of the p-norm as the distance function in the 1-nearest neighbor classifier.

Adding $(\vec{x}', y_{\text{orig}})$ to the training data $\mathcal{D}$ implies that $\vec{x}'$ is no longer the solution to Eq. (5). Therefore, the new closest sample on the decision boundary must have a larger distance to $\vec{x}_{\text{orig}}$ than $\vec{x}'$, otherwise it would have been $\vec{x}'$ before! □ □

B.2 Proof of Theorem 2

Proof. Sketch: The construction of $\mathcal{D}_{\text{poison}}$ contains all samples inside the margin but are classified as -1.

$$-\vec{w}^\top \vec{x} - b \geq 1 - \xi, \ \xi \in (0, 1) \tag{6}$$

$\mathcal{D}_{\text{poison}} \cup \mathcal{D}_{\text{orig}}$ remains linearly separable but the maximum margin shrinks. It follows that the cost of recourse for all $(\vec{x}, h(\vec{x}) = -1)$ increases. □ □

B.3 Proof of Corollary 1

Proof. Follows directly from Theorem 1. □

C Experiments

C.1 Computational Resources

The experiments were run on a machine with 12 Intel(R) Xeon(R) W-2133 CPU @ 3.60GHz cores and 32GB of working memory. The experiments took a bit less than two days to complete.

C.2 Details on the Classifiers

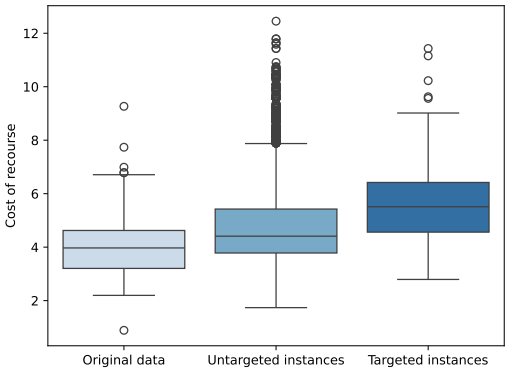
- **RandomForest:** 10 decision tree classifiers each with a maximum depth of 7.
- **DNN:** 3-layer neural network with ReLU activation functions.

C.3 Local Poisoning Attack

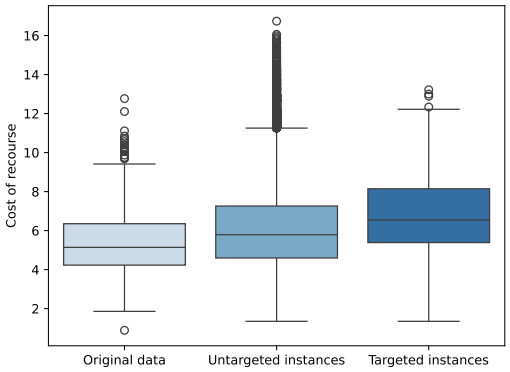
Classifier	Data set	Nearest ↑	DiCE ↑	FACE ↑	Proto ↑
DNN	Diabetes	1.59	1.42	1.24	1.89

Table 3: Difference in the cost of recourse: no vs. local poisoning. Positive numbers denote an increase in the cost of recourse. We report the median (over all folds) rounded to two decimal places.

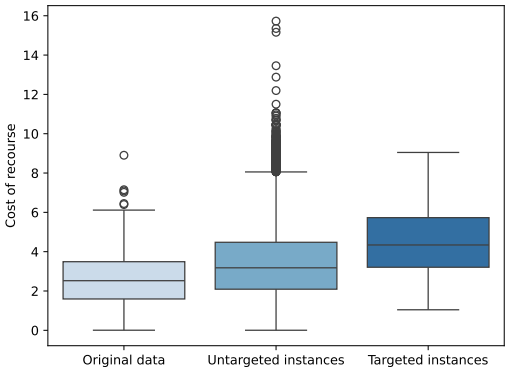
(a) CF: *Nearest*



(b) CF: *DiCE*



(c) CF: *Proto*



(d) CF: *FACE*

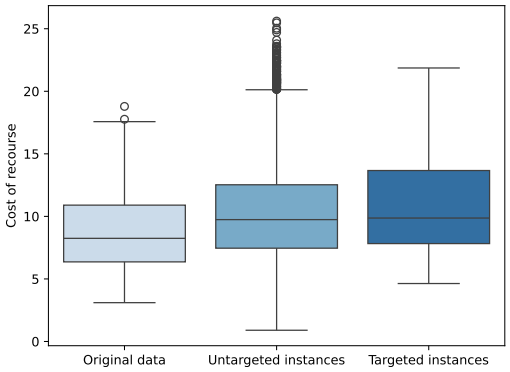
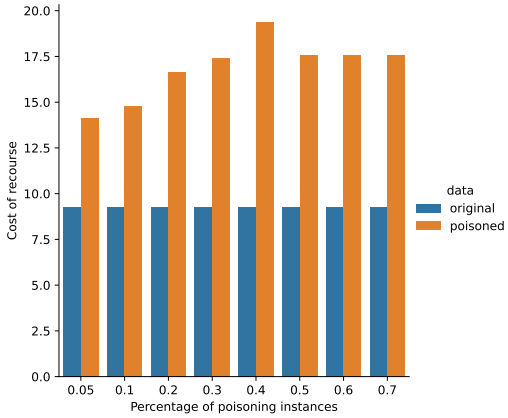


Figure 4: *Local* data poisoning: Cost of recourse (over all test samples) in the case of the diabetes data set and a DNN classifier. Cost of recourse without any data poisoning, of untargeted instances and targeted instances in a local data poisoning.

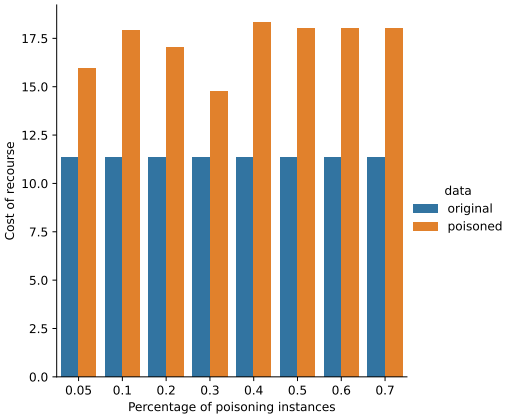
C.4 Sub-group Poisoning Attack

(a) $\mathcal{D}$: Crime – $h(\cdot)$: DNN – CF: *DiCE*

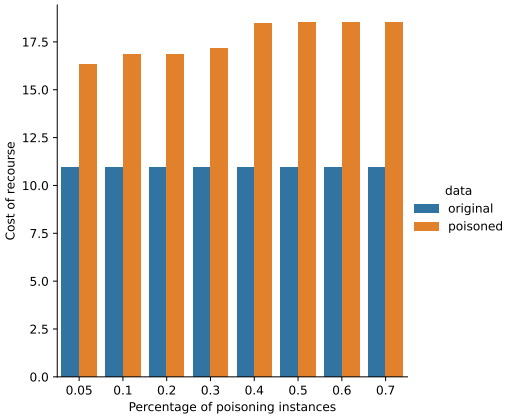


(b) $\mathcal{D}$: Crime – $h(\cdot)$: RNF – CF: *DiCE*

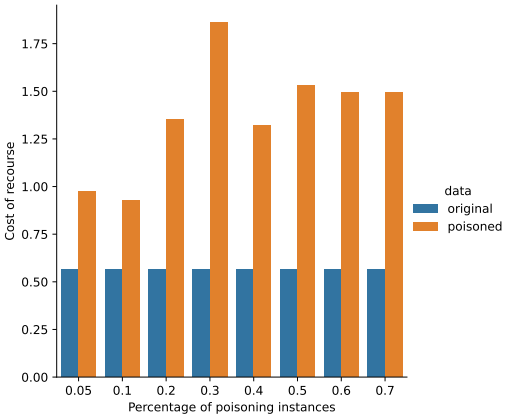
(c) $\mathcal{D}$: Crime – $h(\cdot)$: SVC – CF: *DiCE*

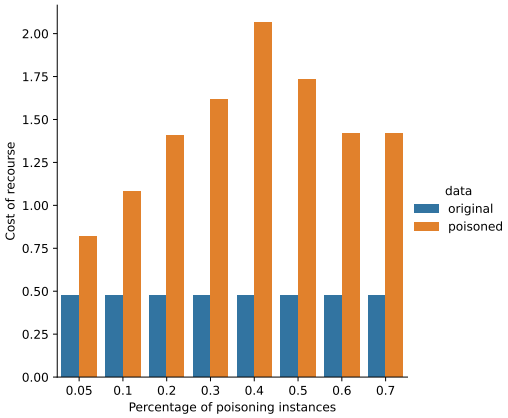


(d) $\mathcal{D}$: Diabetes – $h(\cdot)$: DNN – CF: *DiCE*

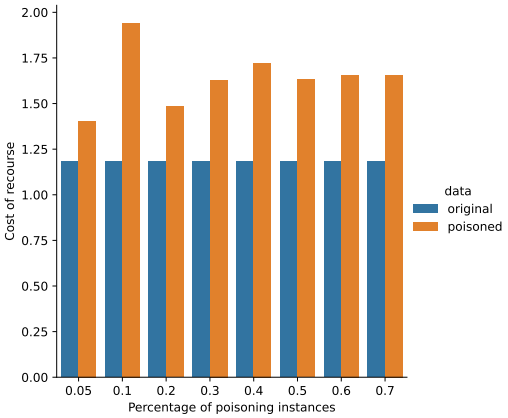


(e) $\mathcal{D}$: Diabetes – $h(\cdot)$: RNF – CF: *DiCE*

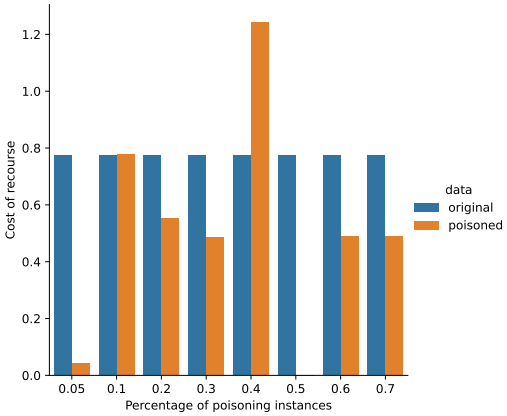




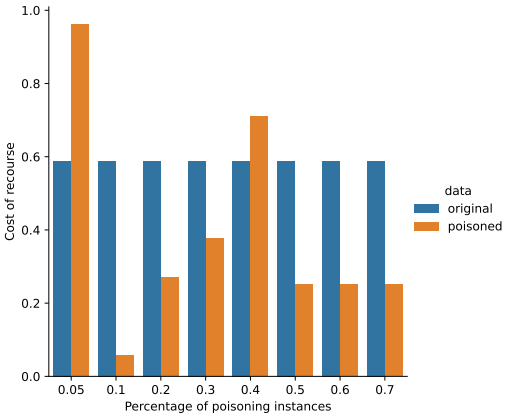
(a) $\mathcal{D}$: Diabetes – $h(\cdot)$: SVC – CF: *DiCE*



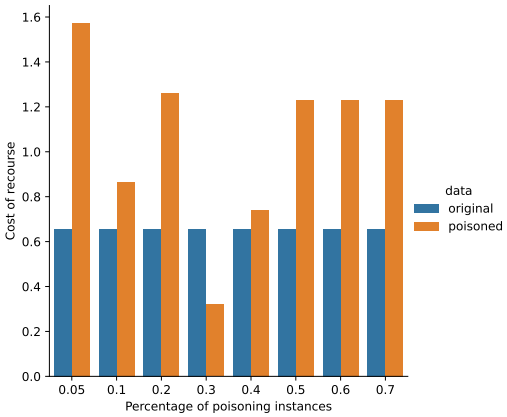
(b) $\mathcal{D}$: Credit – $h(\cdot)$: DNN – CF: *DiCE*



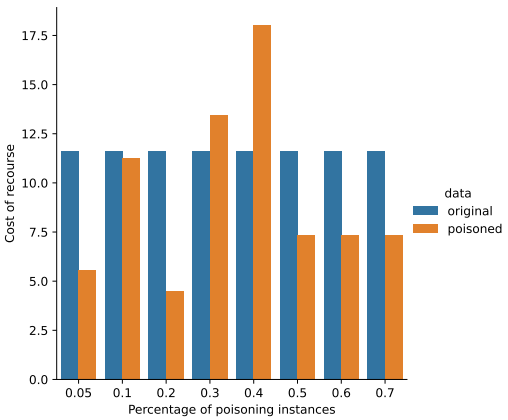
(c) $\mathcal{D}$: Credit – $h(\cdot)$: RNF – CF: *DiCE*



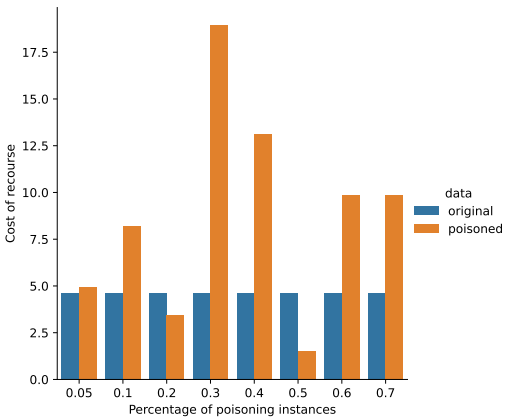
(d) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *DiCE*



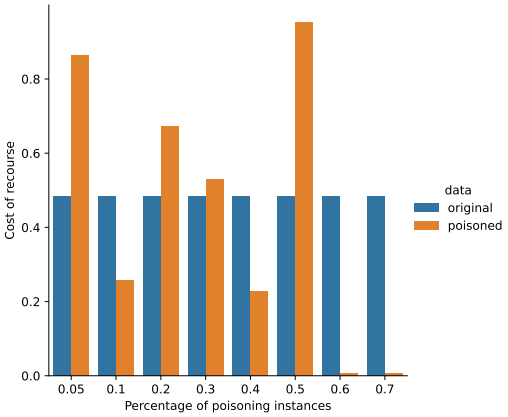
(e) $\mathcal{D}$: Crime – $h(\cdot)$: DNN – CF: *FACE*



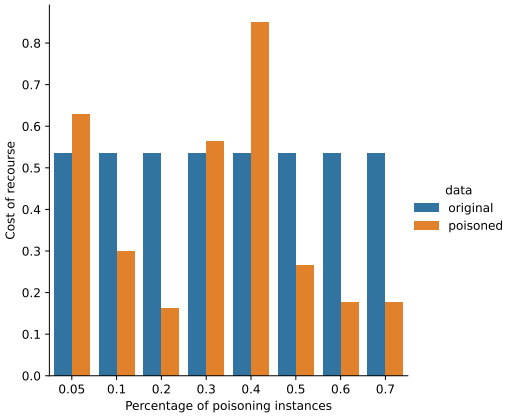
(f) $\mathcal{D}$: Crime – $h(\cdot)$: RNF – CF: *FACE*



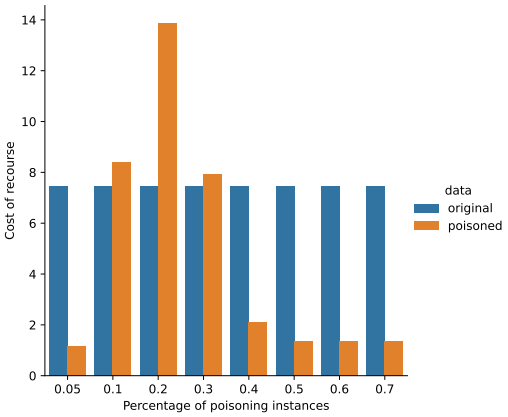
(a) $\mathcal{D}$: Diabetes – $h(\cdot)$: RNF – CF: *FACE*



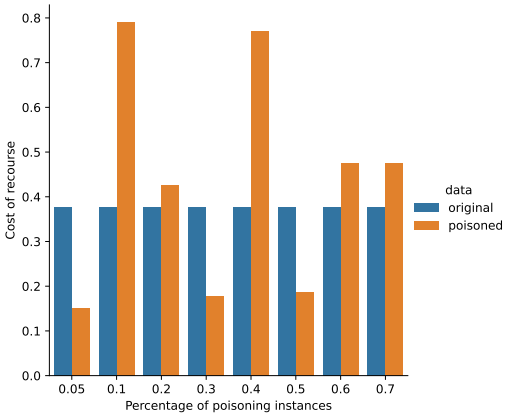
(b) $\mathcal{D}$: Diabetes – $h(\cdot)$: SVC – CF: *FACE*



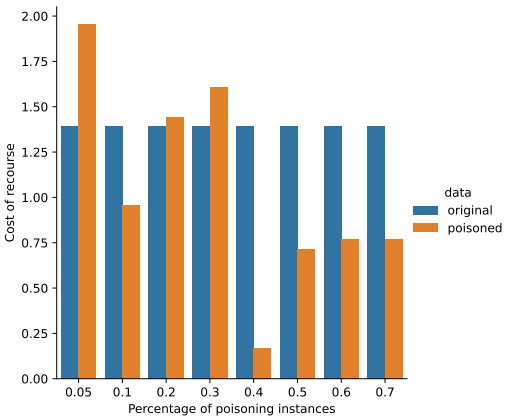
(c) $\mathcal{D}$: Crime – $h(\cdot)$: SVC – CF: *FACE*



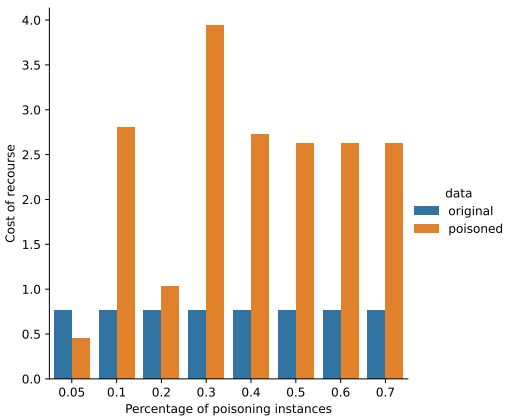
(d) $\mathcal{D}$: Diabetes – $h(\cdot)$: DNN – CF: *FACE*



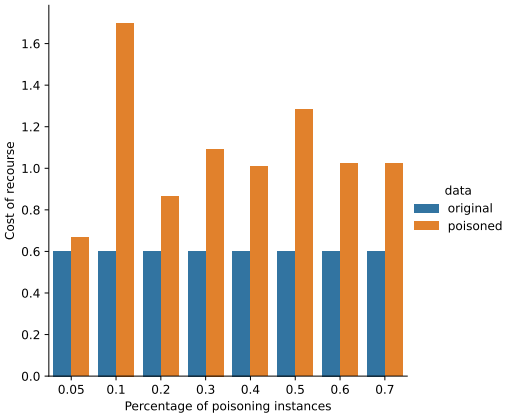
(e) $\mathcal{D}$: Credit – $h(\cdot)$: DNN – CF: *FACE*



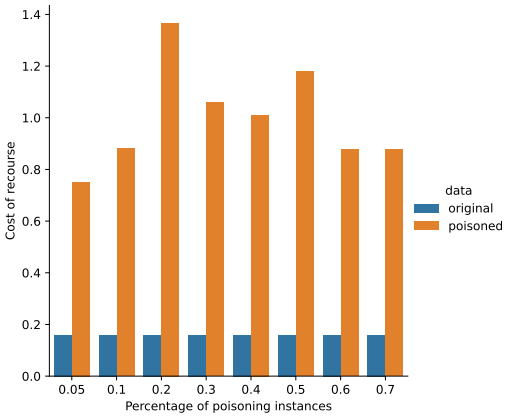
(f) $\mathcal{D}$: Credit – $h(\cdot)$: RNF – CF: *FACE*



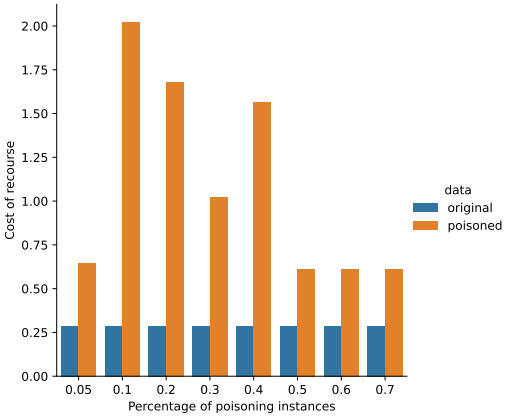
(a) $\mathcal{D}$: Diabetes – $h(\cdot)$: DNN – CF: *Nearest*



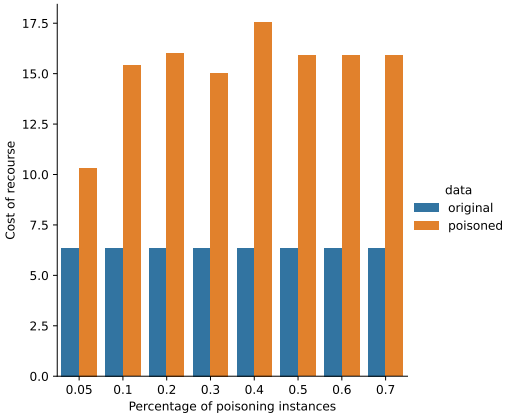
(b) $\mathcal{D}$: Diabetes – $h(\cdot)$: RNF – CF: *Nearest*



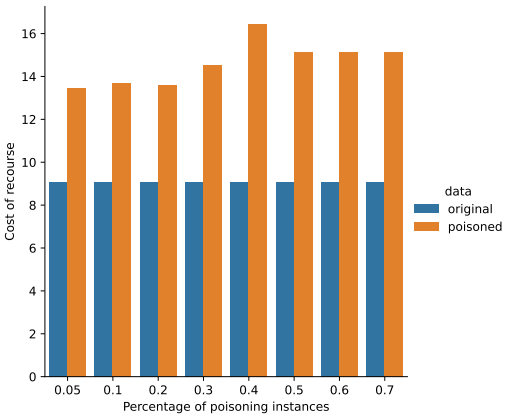
(c) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *FACE*



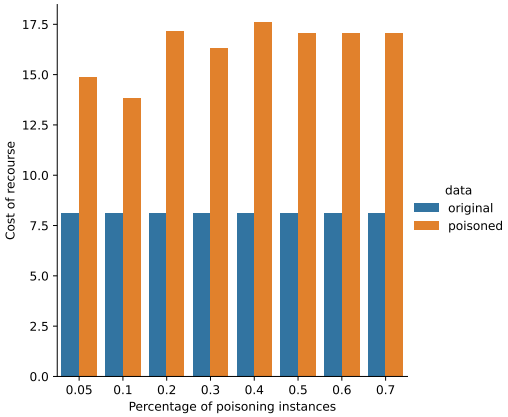
(d) $\mathcal{D}$: Crime – $h(\cdot)$: DNN – CF: *Nearest*



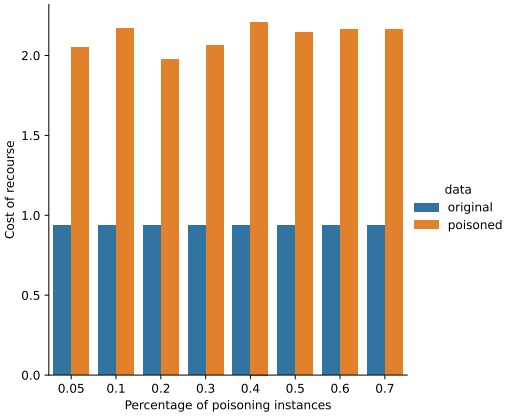
(e) $\mathcal{D}$: Crime – $h(\cdot)$: RNF – CF: *Nearest*



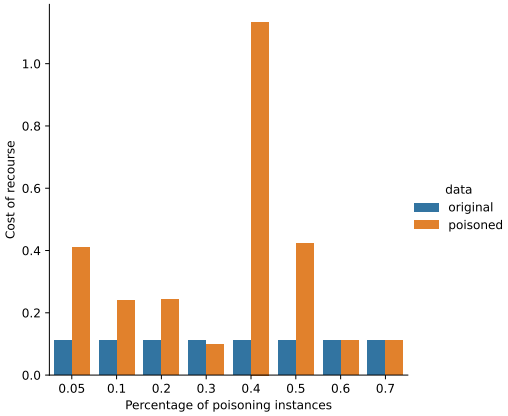
(f) $\mathcal{D}$: Crime – $h(\cdot)$: SVC – CF: *Nearest*



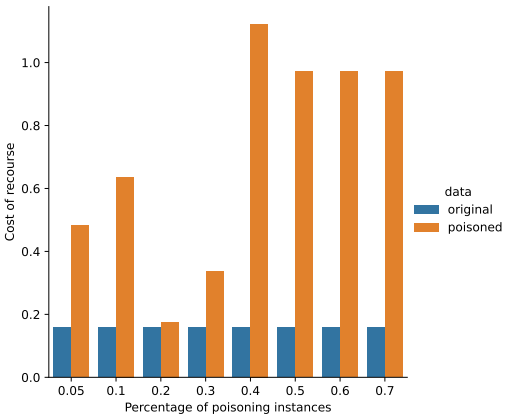
(a) $\mathcal{D}$: Diabetes – $h(\cdot)$: SVC – CF: *Nearest*



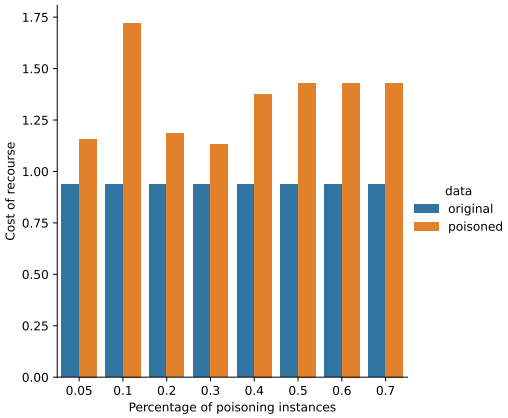
(b) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *Nearest*



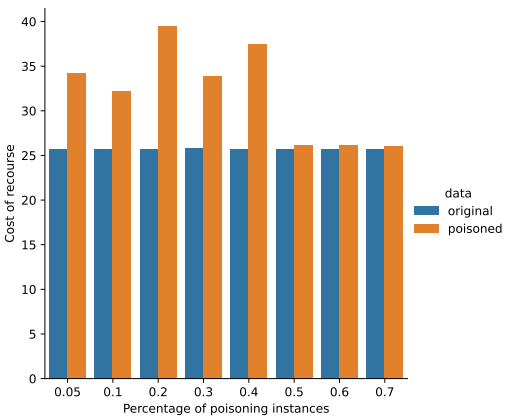
(c) $\mathcal{D}$: Credit – $h(\cdot)$: RNF – CF: *Nearest*



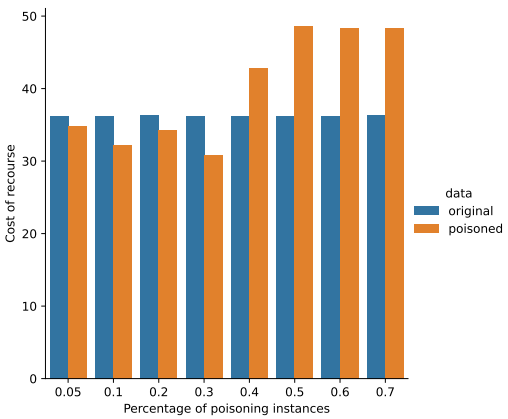
(d) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *Nearest*



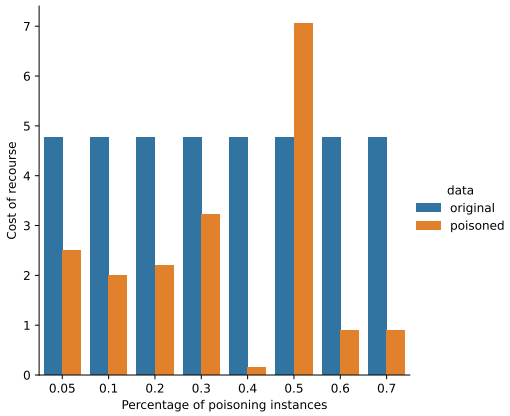
(e) $\mathcal{D}$: Crime – $h(\cdot)$: DNN – CF: *Proto*



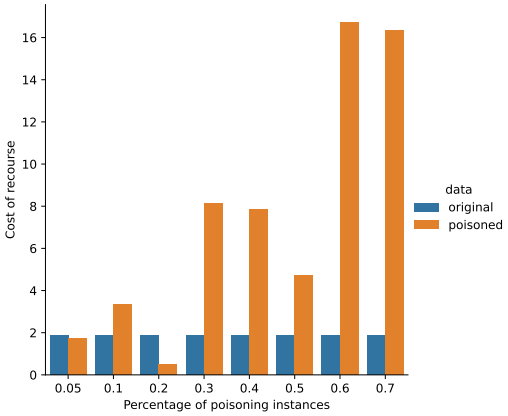
(f) $\mathcal{D}$: Crime – $h(\cdot)$: SVC – CF: *Proto*



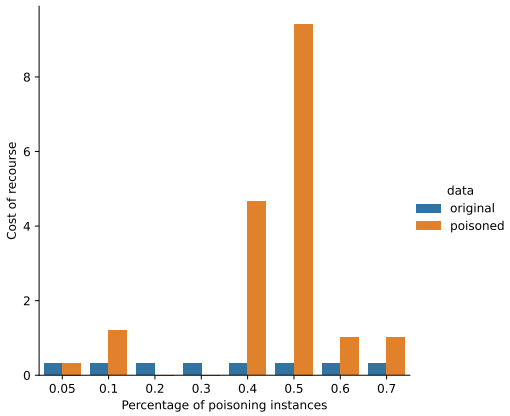
(a) $\mathcal{D}$: Diabetes – $h(\cdot)$: DNN – CF: *Proto*



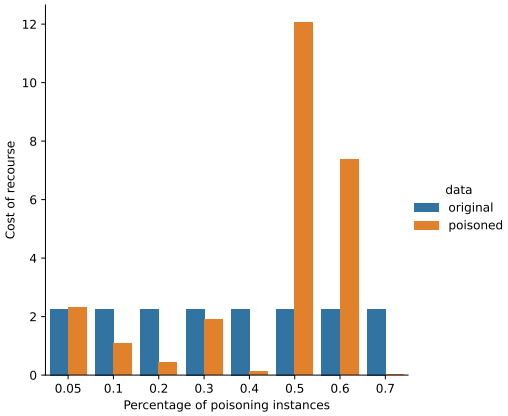
(b) $\mathcal{D}$: Diabetes – $h(\cdot)$: SVC – CF: *Proto*



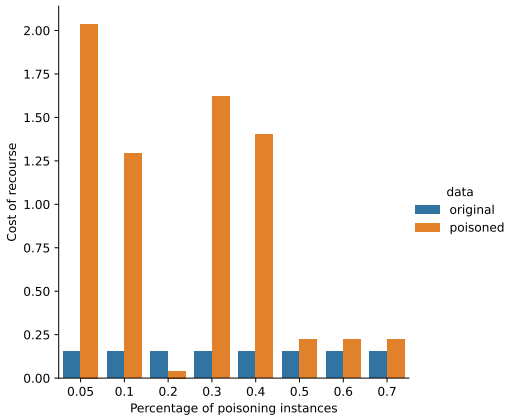
(c) $\mathcal{D}$: Credit – $h(\cdot)$: DNN – CF: *Proto*



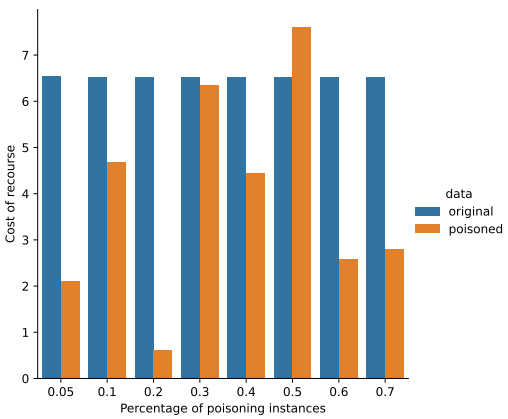
(d) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *Proto*



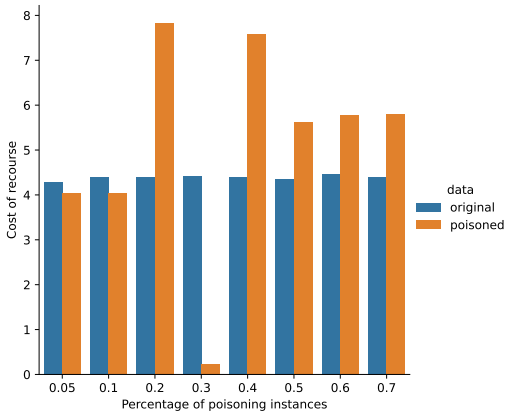
(e) $\mathcal{D}$: Credit – $h(\cdot)$: RNF – CF: *Proto*



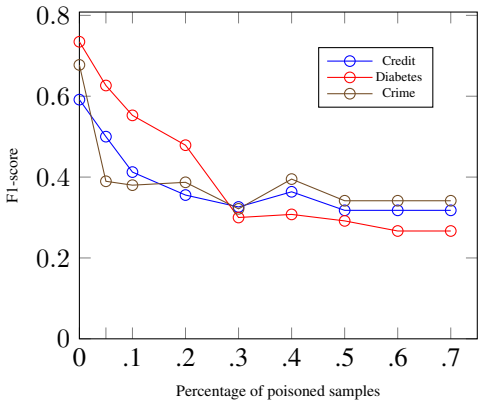
(f) $\mathcal{D}$: Diabetes – $h(\cdot)$: RNF – CF: *Proto*



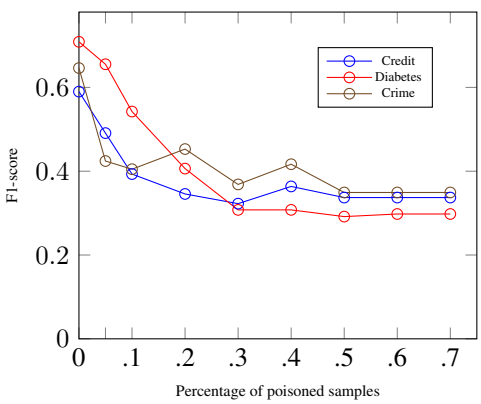
(a) $\mathcal{D}$: Crime – $h(\cdot)$: RNF – CF: *Proto*



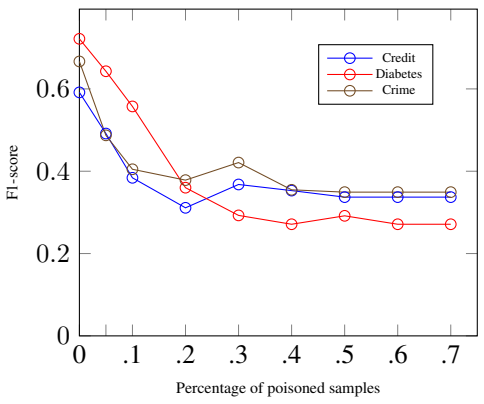
(a) $h(\cdot)$: SVC – CF: *Nearest*



(b) $h(\cdot)$: SVC – CF: *DiCE*



(c) $h(\cdot)$: SVC – CF: *FACE*



(d) $h(\cdot)$: SVC – CF: *Proto*

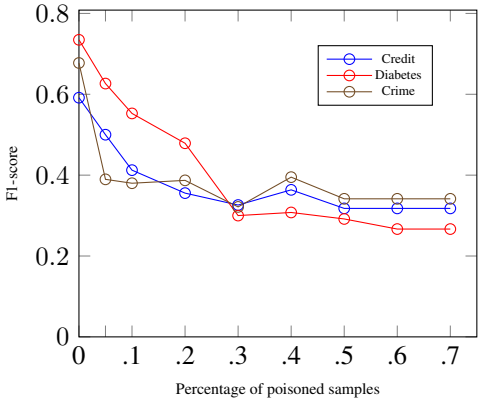


Figure 12: Sub-group data poisoning attack: Median (over all folds) F1-score of the classifier for different percentages of poisoned samples (0% to 70%).

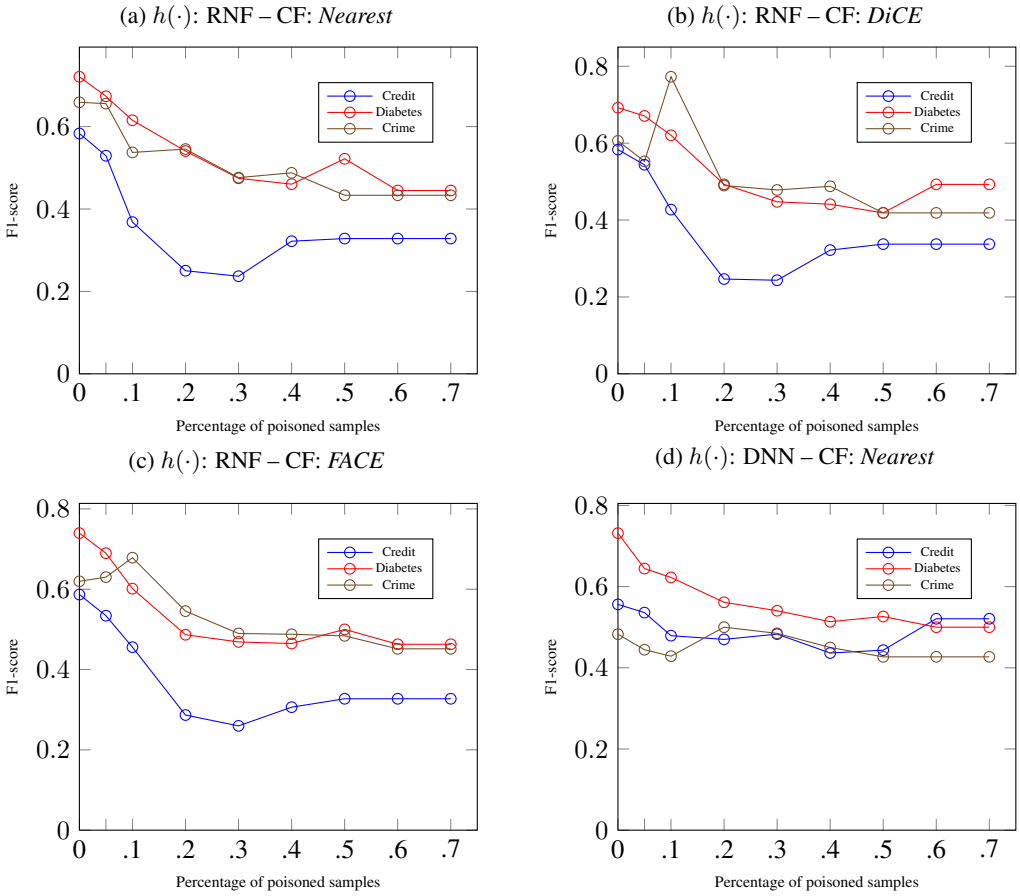
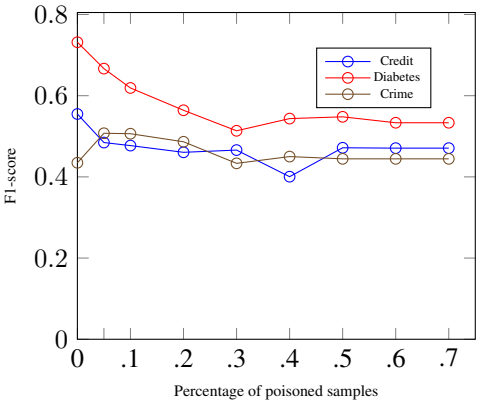
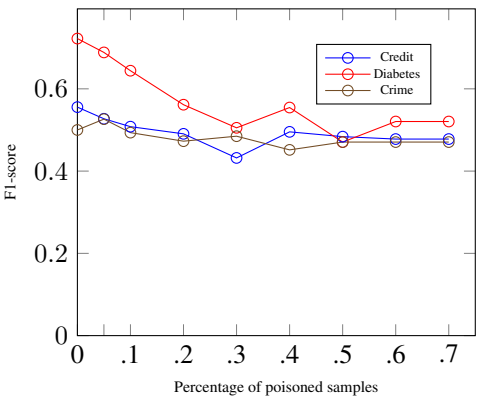


Figure 13: Sub-group data poisoning attack: Median (over all folds) F1-score of the classifier for different percentages of poisoned instances (0% to 70%).

(a) $h(\cdot)$: DNN – CF: *DiCE*



(b) $h(\cdot)$: DNN – CF: *FACE*



(c) $h(\cdot)$: DNN – CF: *Proto*

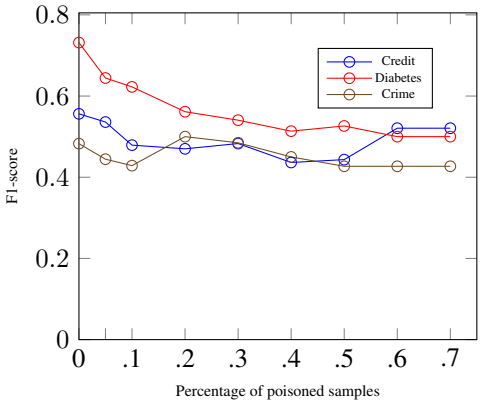


Figure 14: Sub-group data poisoning attack: Median (over all folds) F1-score of the classifier for different percentages of poisoned instances (0% to 70%).

C.5 Global Poisoning Attack

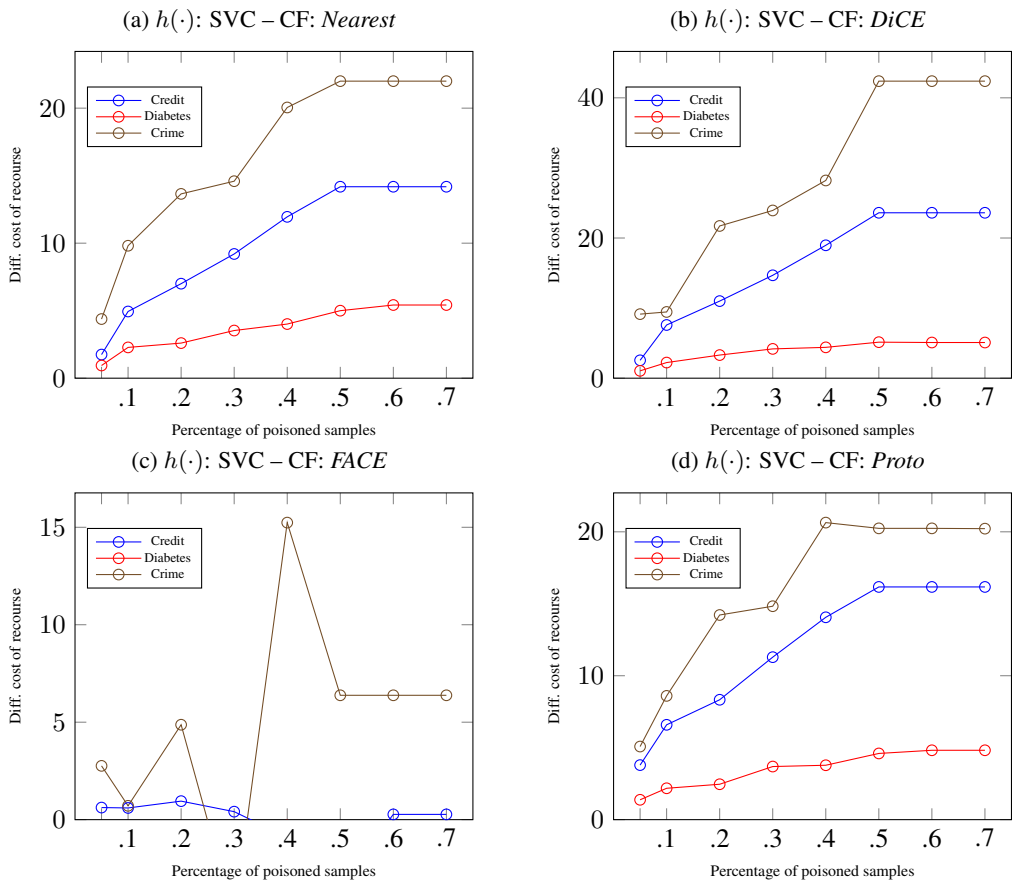
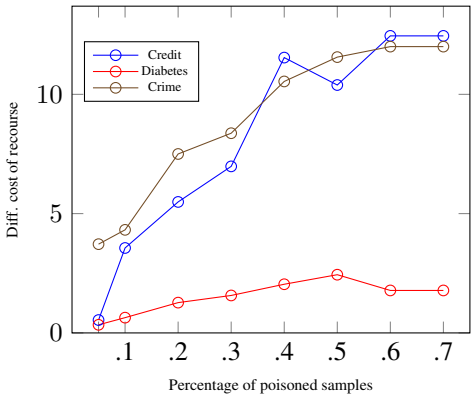
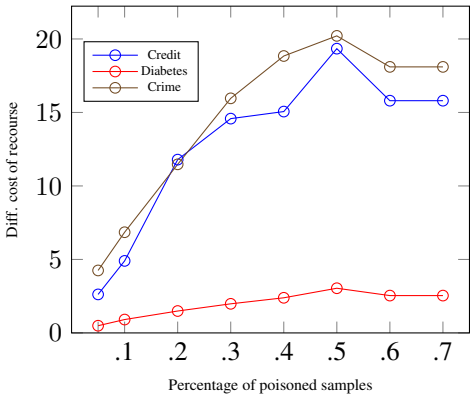


Figure 15: Global data poisoning attack: Median (over all folds) difference in the cost of recourse vs. percentage of poisoned instances (5% to 70%).

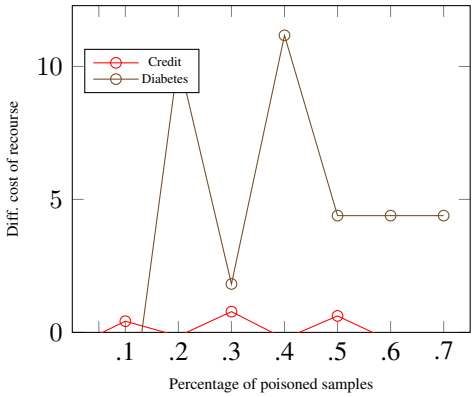
(a) $h(\cdot)$: RNF – CF: *Nearest*



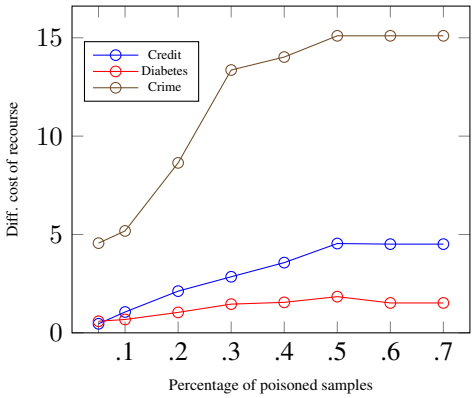
(b) $h(\cdot)$: RNF – CF: *DiCE*



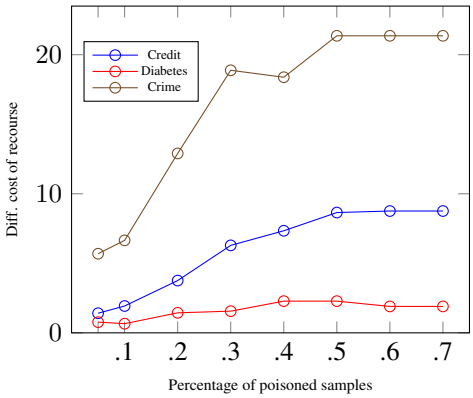
(c) $h(\cdot)$: RNF – CF: *FACE*



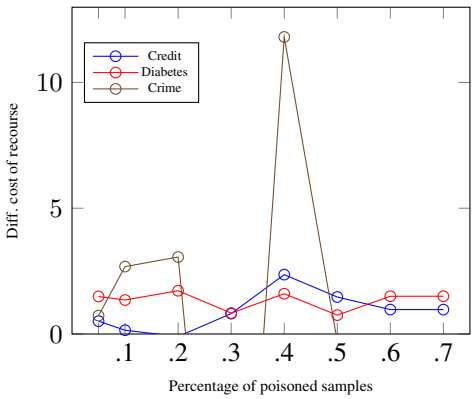
(d) $h(\cdot)$: DNN – CF: *Nearest*



(e) $h(\cdot)$: DNN – CF: *DiCE*



(a) $h(\cdot)$: DNN – CF: *FACE*



(b) $h(\cdot)$: DNN – CF: *Proto*

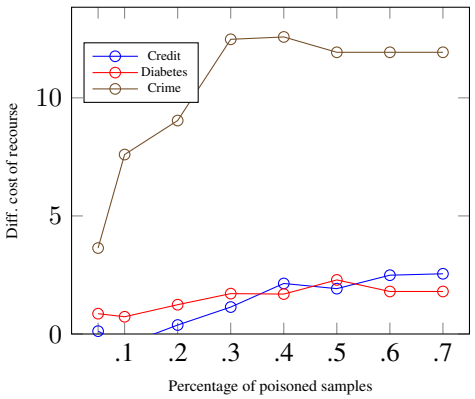
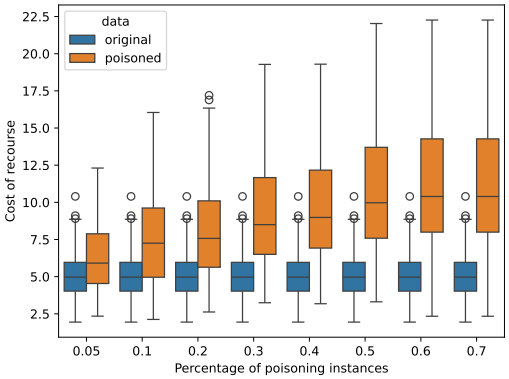
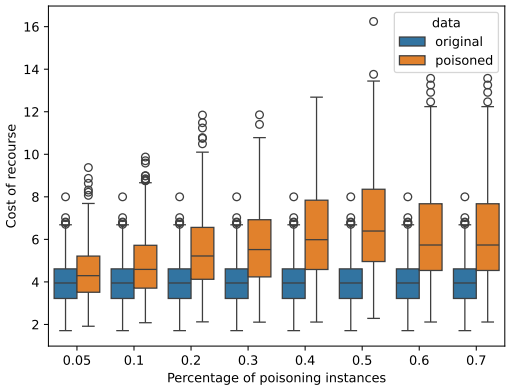


Figure 17: Global data poisoning attack: Median (over all folds) difference in the cost of recourse vs. percentage of poisoned instances (5% to 70%).

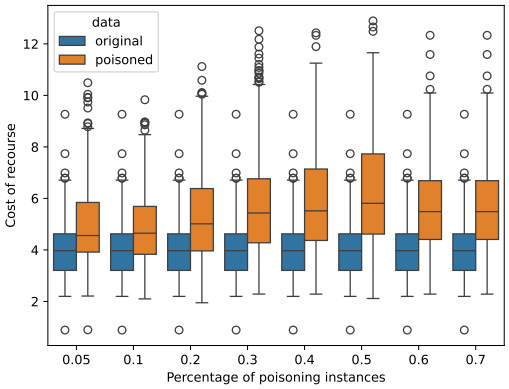
(a) $\mathcal{D}$: Diabetes – $h(\cdot)$: SVC – CF: *Nearest*



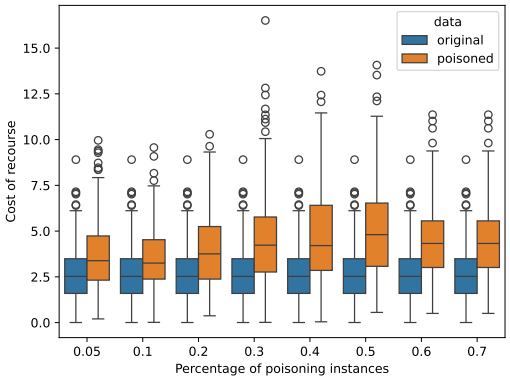
(b) $\mathcal{D}$: Diabetes – $h(\cdot)$: RNF – CF: *Nearest*



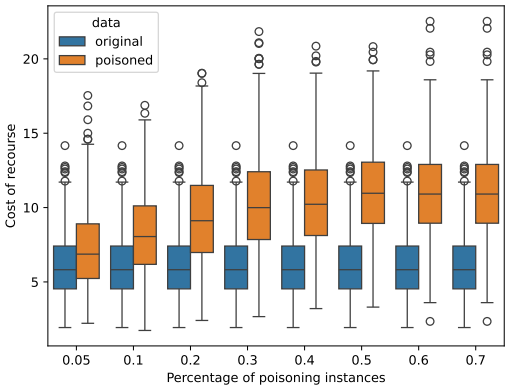
(c) $\mathcal{D}$: Diabetes – $h(\cdot)$: DNN – CF: *Nearest*



(d) $\mathcal{D}$: Diabetes – $h(\cdot)$: DNN – CF: *Proto*



(e) $\mathcal{D}$: Diabetes – $h(\cdot)$: SVC – CF: *DiCE*



(f) $\mathcal{D}$: Diabetes – $h(\cdot)$: RNF – CF: *DiCE*

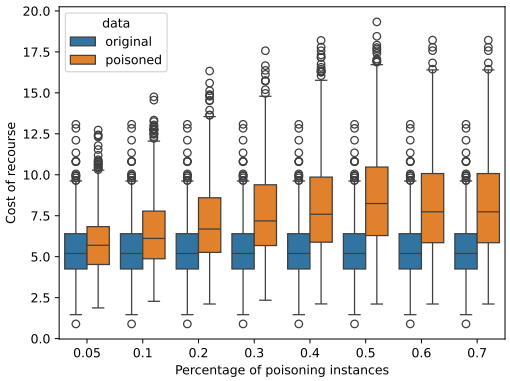
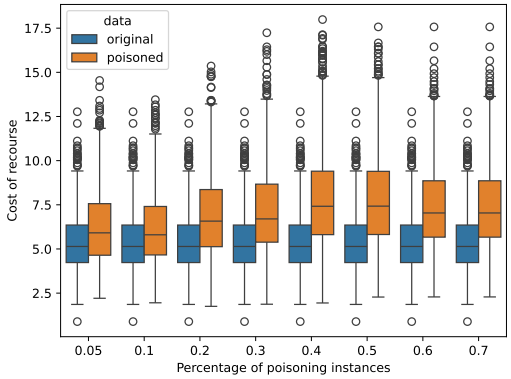


Figure 18: Cost of recourse (over all folds) of original data vs. poisoned data – (5% to 70% of poisoned instances).

(a) $\mathcal{D}$: Diabetes – $h(\cdot)$: DNN – CF: *DiCE*



(b) $\mathcal{D}$: Diabetes – $h(\cdot)$: SVC – CF: *FACE*

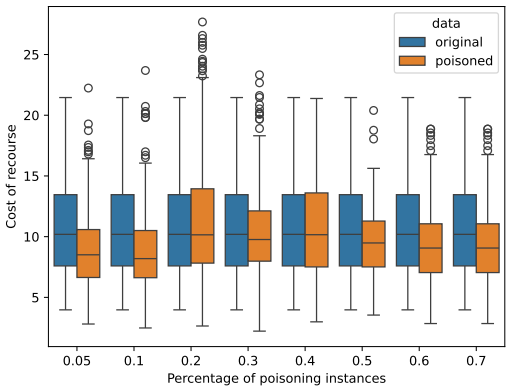
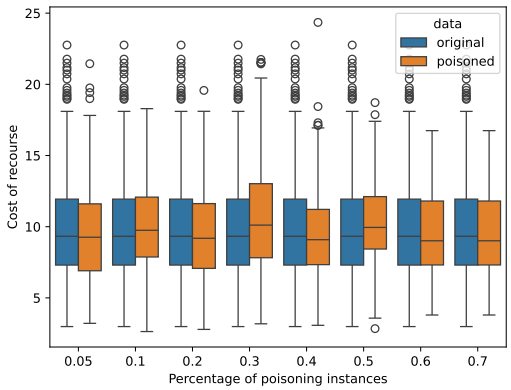
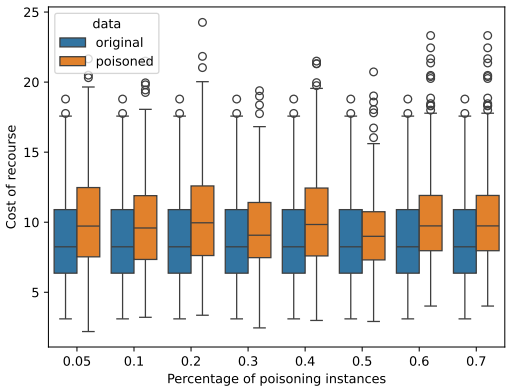


Figure 19: Cost of recourse (over all folds) of original data vs. poisoned data – (5% to 70% of poisoned instances).

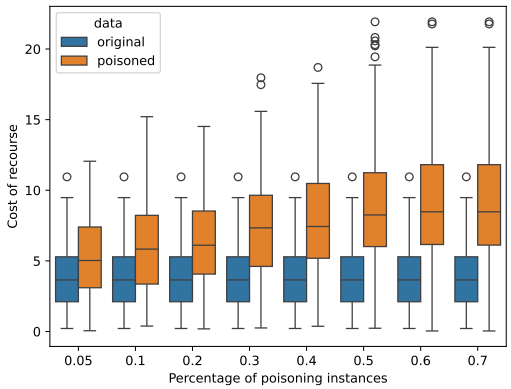
(a) $\mathcal{D}$: Diabetes – $h(\cdot)$: RNF – CF: *FACE*



(b) $\mathcal{D}$: Diabetes – $h(\cdot)$: DNN – CF: *FACE*



(c) $\mathcal{D}$: Diabetes – $h(\cdot)$: SVC – CF: *Proto*



(d) $\mathcal{D}$: Diabetes – $h(\cdot)$: RNF – CF: *Proto*

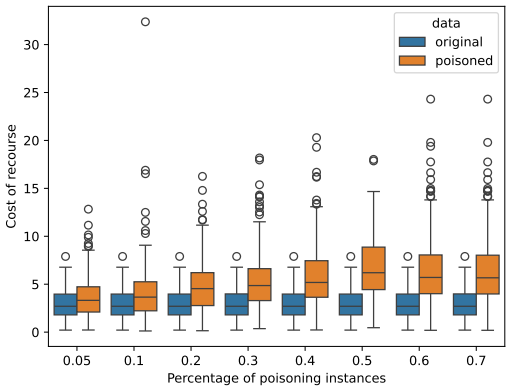
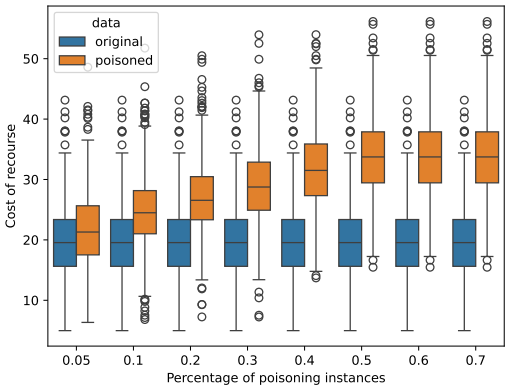
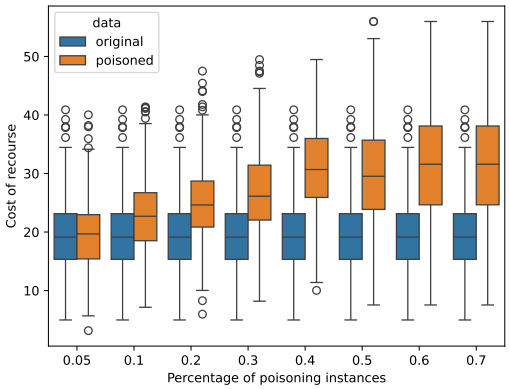


Figure 20: Cost of recourse (over all folds) of original data vs. poisoned data – (5% to 70% of poisoned instances).

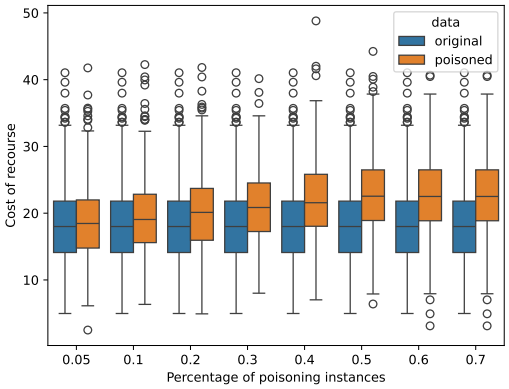
(a) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *Nearest*



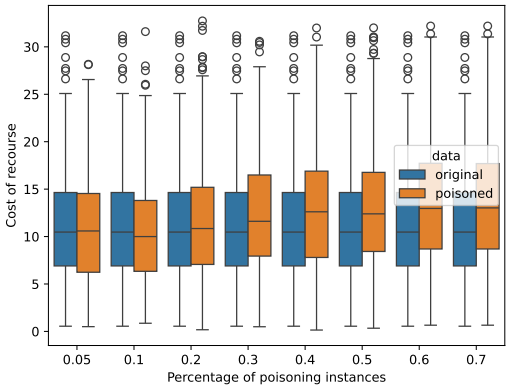
(b) $\mathcal{D}$: Credit – $h(\cdot)$: RNF – CF: *Nearest*



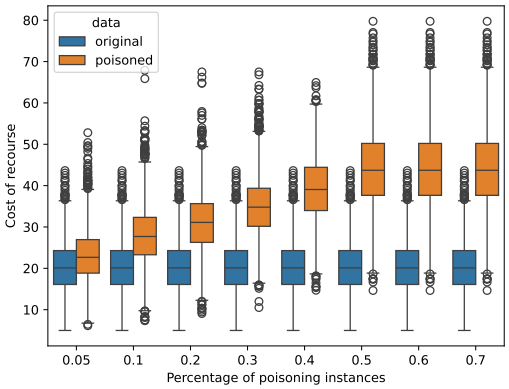
(c) $\mathcal{D}$: Credit – $h(\cdot)$: DNN – CF: *Nearest*



(d) $\mathcal{D}$: Credit – $h(\cdot)$: DNN – CF: *Proto*



(e) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *DiCE*



(f) $\mathcal{D}$: Credit – $h(\cdot)$: RNF – CF: *DiCE*

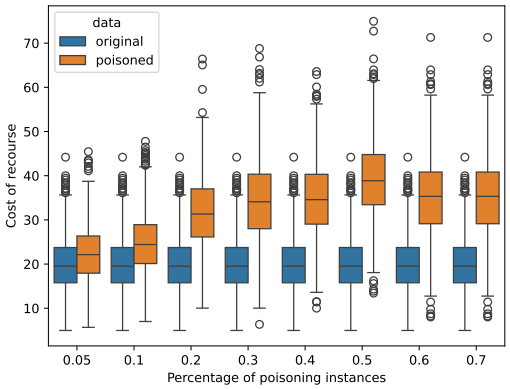
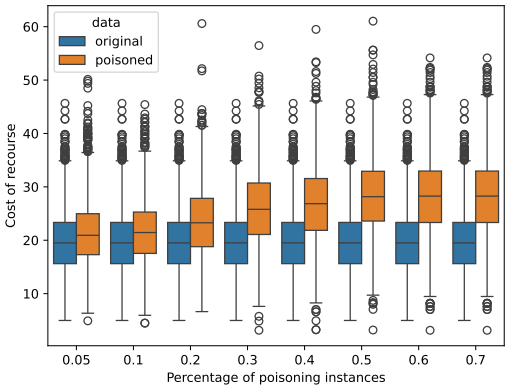


Figure 21: Cost of recourse (over all folds) of original data vs. poisoned data – (5% to 70% of poisoned instances).

(a) $\mathcal{D}$: Credit – $h(\cdot)$: DNN – CF: *DiCE*



(b) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *FACE*

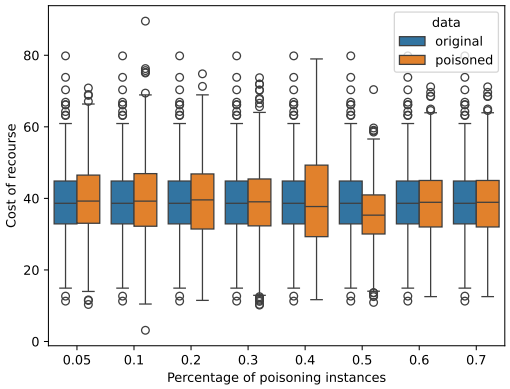
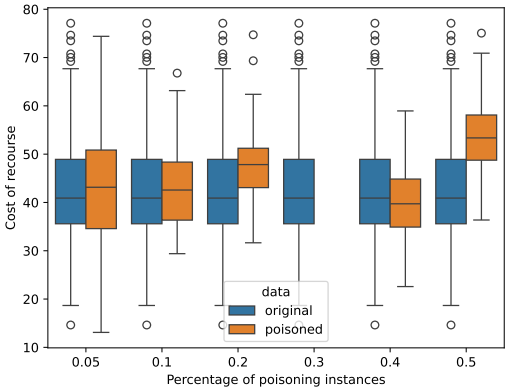
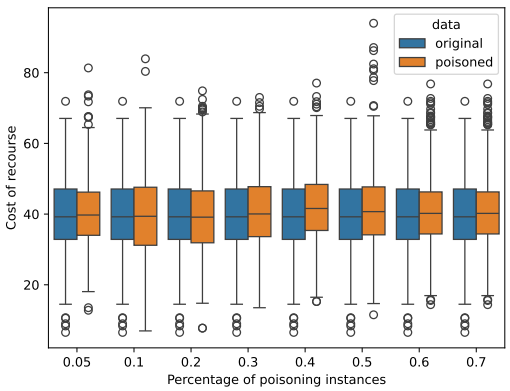


Figure 22: Cost of recourse (over all folds) of original data vs. poisoned data – (5% to 70% of poisoned instances).

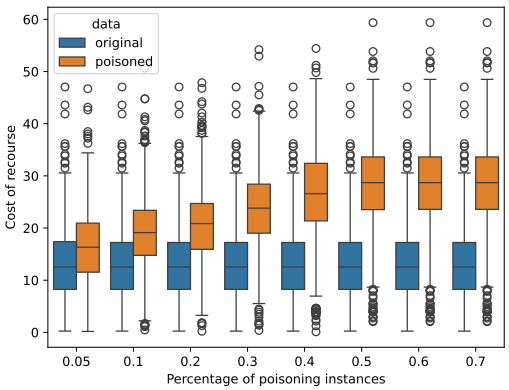
(a) $\mathcal{D}$: Credit – $h(\cdot)$: RNF – CF: *FACE*



(b) $\mathcal{D}$: Credit – $h(\cdot)$: DNN – CF: *FACE*



(c) $\mathcal{D}$: Credit – $h(\cdot)$: SVC – CF: *Proto*



(d) $\mathcal{D}$: Credit – $h(\cdot)$: RNF – CF: *Proto*

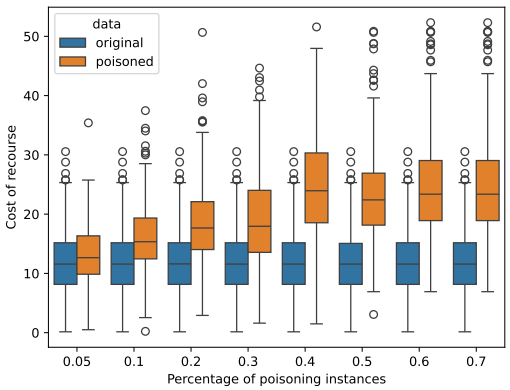
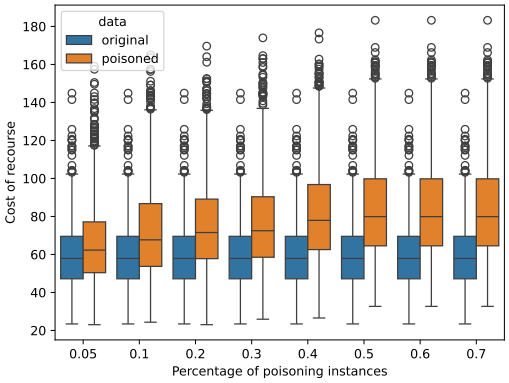
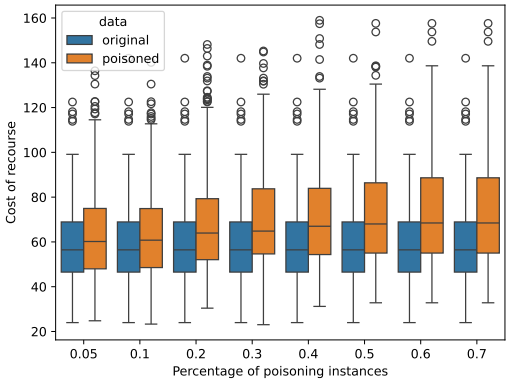


Figure 23: Cost of recourse (over all folds) of original data vs. poisoned data – (5% to 70% of poisoned instances).

(a) $\mathcal{D}$: Crime – $h(\cdot)$: SVC – CF: *Nearest*

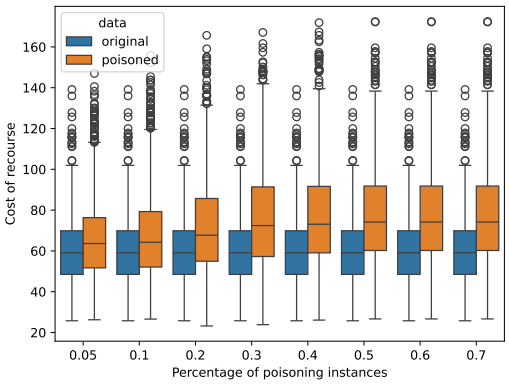


(b) $\mathcal{D}$: Crime – $h(\cdot)$: RNF – CF: *Nearest*

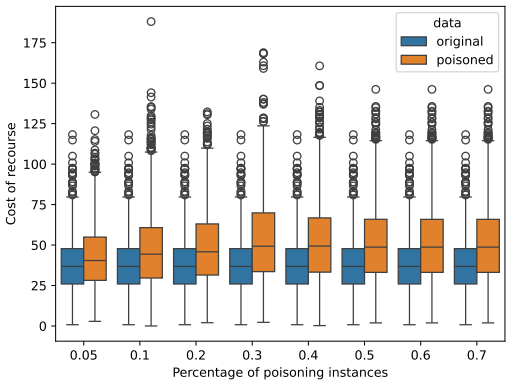


(c) $\mathcal{D}$: Crime – $h(\cdot)$: DNN – CF: *Nearest*

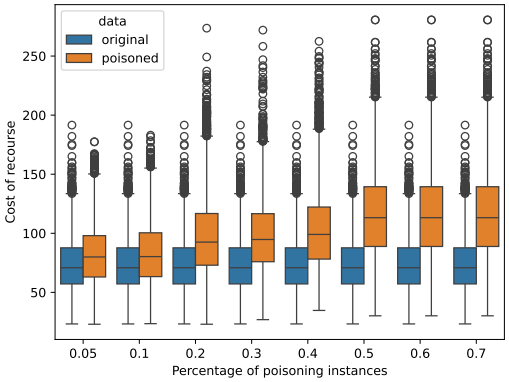
(d) $\mathcal{D}$: Crime – $h(\cdot)$: DNN – CF: *Proto*



(e) $\mathcal{D}$: Crime – $h(\cdot)$: SVC – CF: *DiCE*



(f) $\mathcal{D}$: Crime – $h(\cdot)$: RNF – CF: *DiCE*



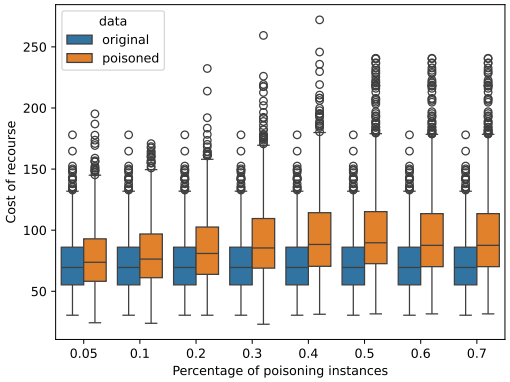
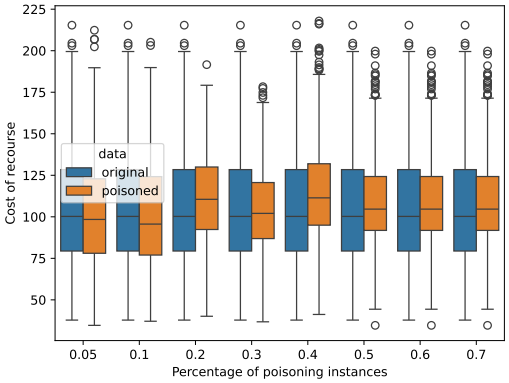
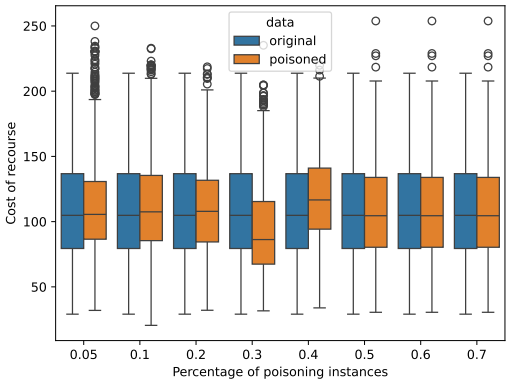


Figure 24: Cost of recourse (over all folds) of original data vs. poisoned data – (5% to 70% of poisoned instances).

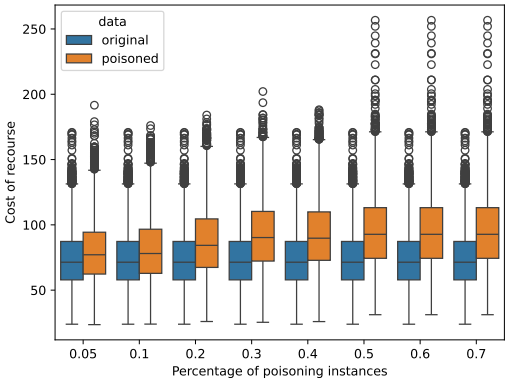
(a) $\mathcal{D}$: Crime – $h(\cdot)$: RNF – CF: *FACE*



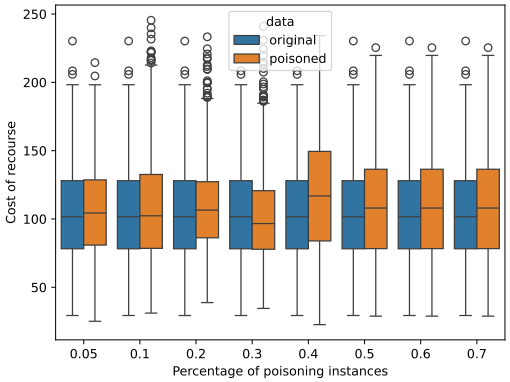
(b) $\mathcal{D}$: Crime – $h(\cdot)$: DNN – CF: *FACE*



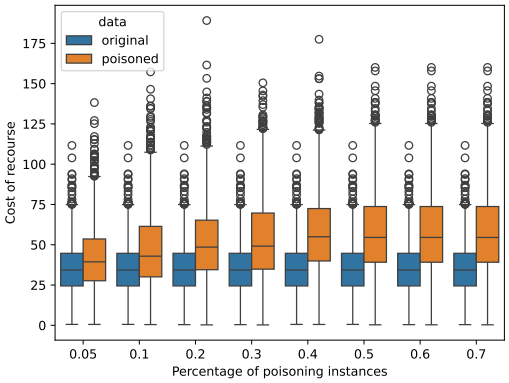
(c) $\mathcal{D}$: Crime – $h(\cdot)$: DNN – CF: *DiCE*



(d) $\mathcal{D}$: Crime – $h(\cdot)$: SVC – CF: *FAE*



(e) $\mathcal{D}$: Crime – $h(\cdot)$: SVC – CF: *Proto*



(f) $\mathcal{D}$: Crime – $h(\cdot)$: RNF – CF: *Proto*

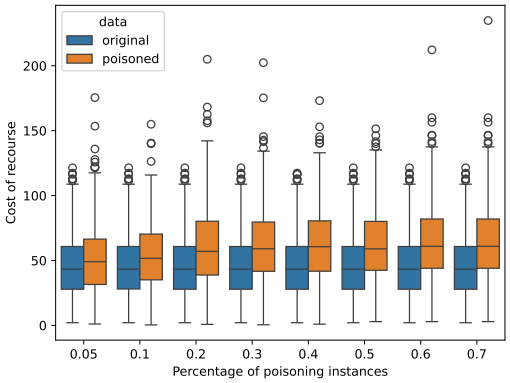


Figure 25: Cost of recourse (over all folds) of original data vs. poisoned data – (5% to 70% of poisoned instances).

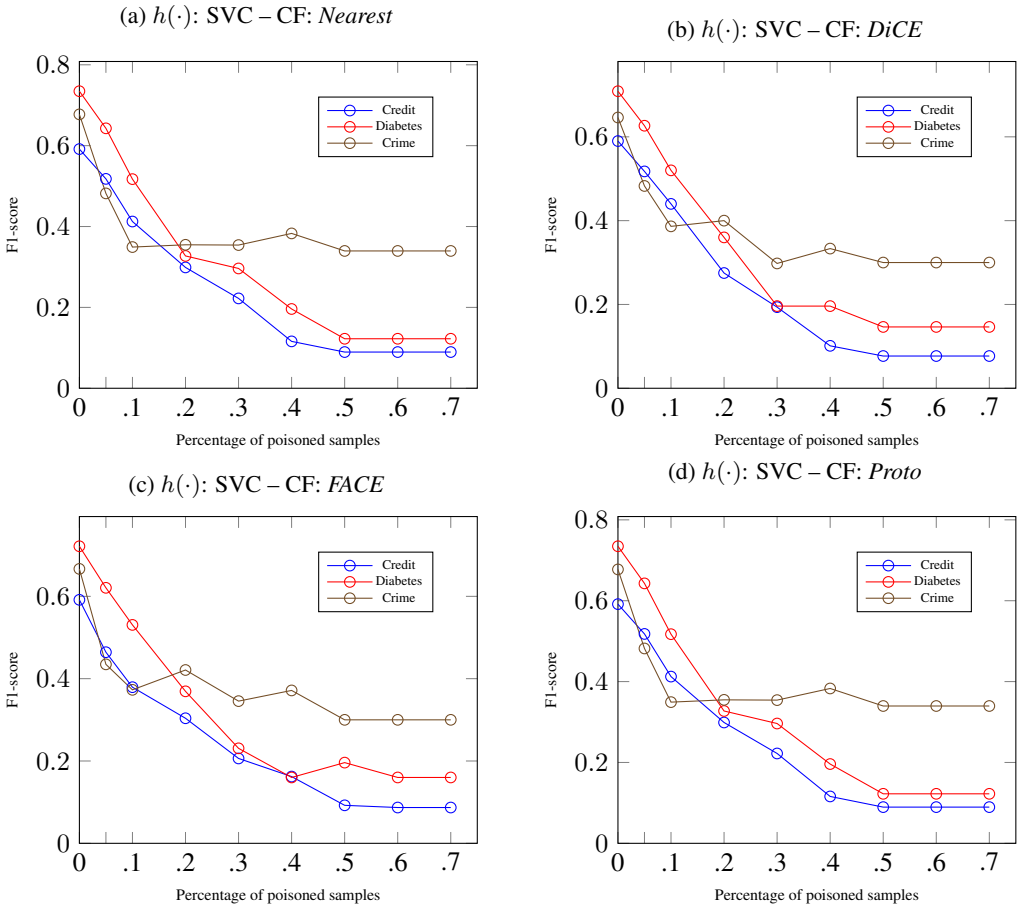
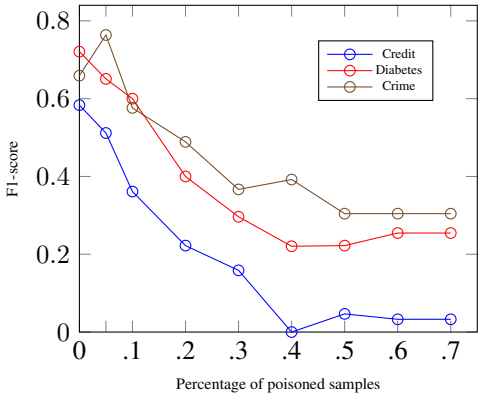
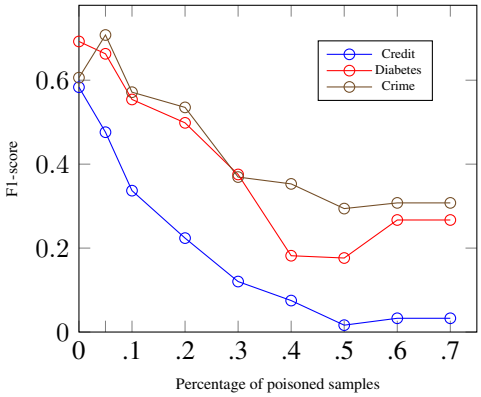


Figure 26: Global data poisoning attack: Median (over all folds) F1-score of the classifier for different percentages of poisoned samples (0% to 70%).

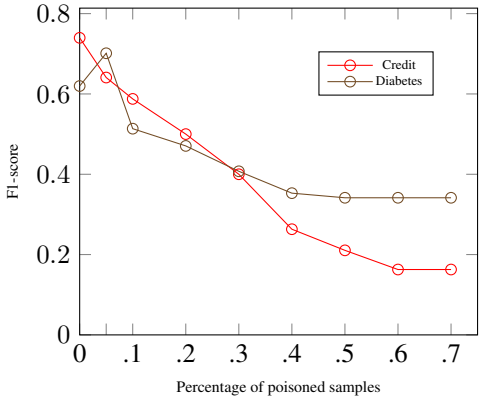
(a) $h(\cdot)$: RNF – CF: *Nearest*



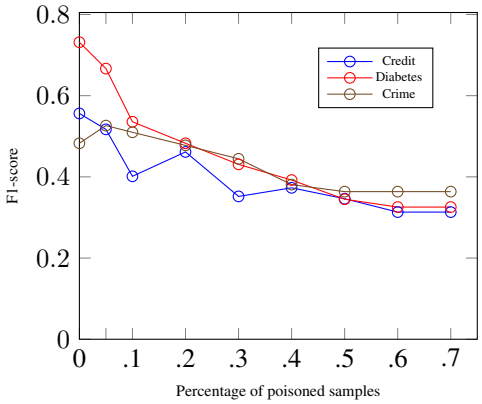
(b) $h(\cdot)$: RNF – CF: *DiCE*



(c) $h(\cdot)$: RNF – CF: *FACE*



(d) $h(\cdot)$: DNN – CF: *Nearest*



(e) $h(\cdot)$: DNN – CF: *DiCE*

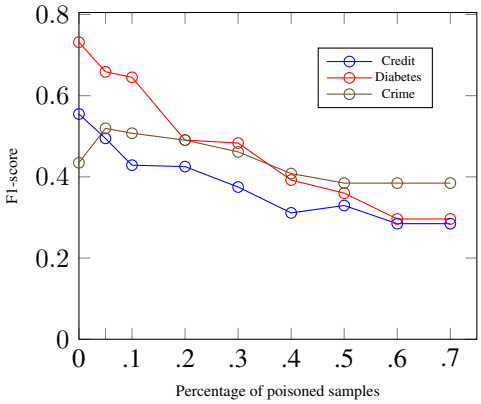
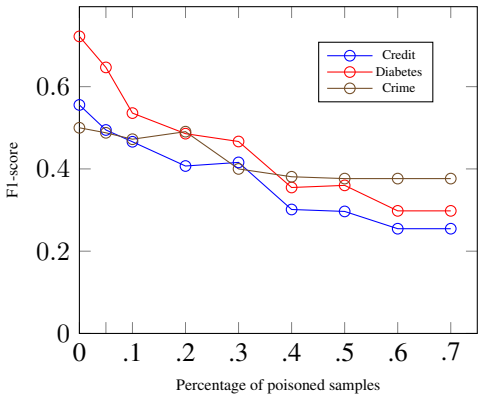


Figure 27: Global data poisoning attack: Median (over all folds) F1-score of the classifier for different percentages of poisoned samples (0% to 70%).

(a) $h(\cdot)$: DNN – CF: *FACE*



(b) $h(\cdot)$: DNN – CF: *Proto*

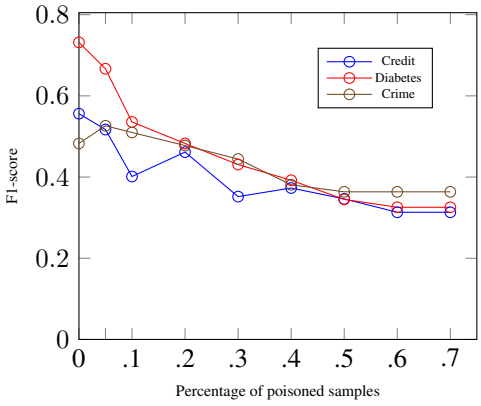
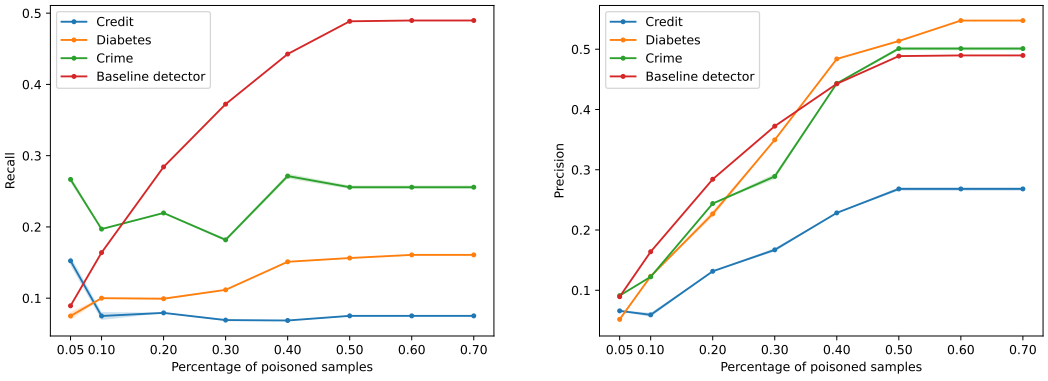


Figure 28: Global data poisoning attack: Median (over all folds) F1-score of the classifier for different percentages of poisoned samples (0% to 70%).

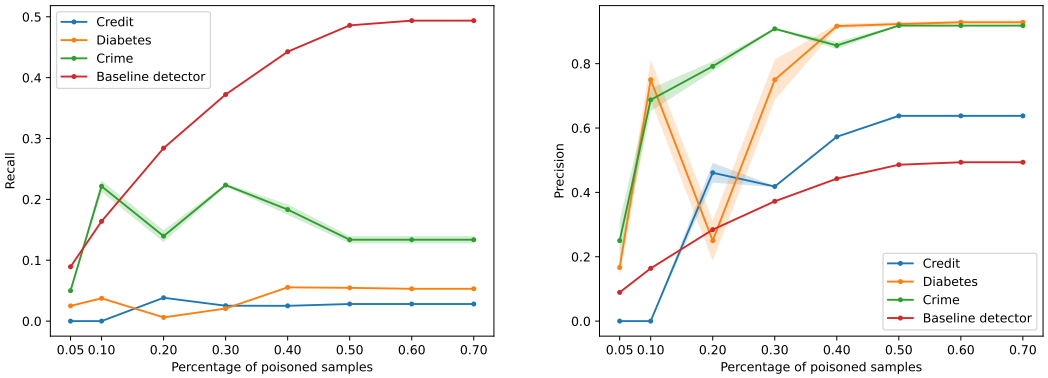
C.6 Detection of Poisonous Instances



(a) Recall

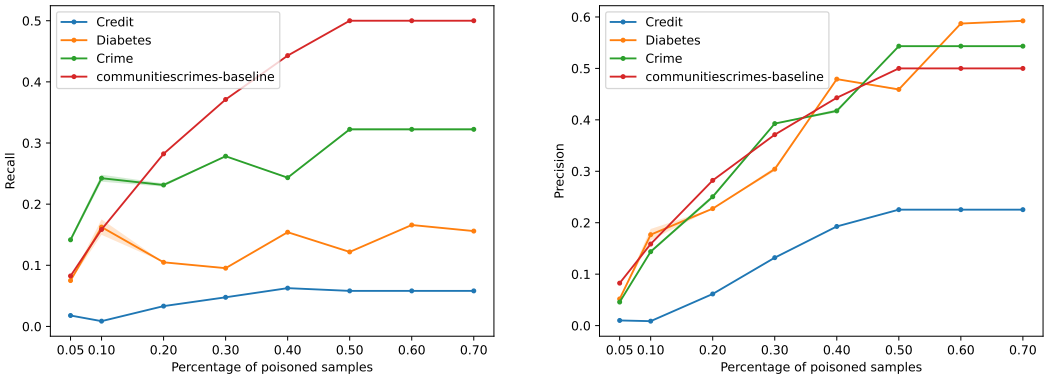
(b) Precision

Figure 29: SVC: Isolation Forest for detection of the poisonous instances in a global poisoning. We report the mean and variance (over all folds) of the recall and precision (larger numbers are better). Note that the variance is not visible because it is close to zero.



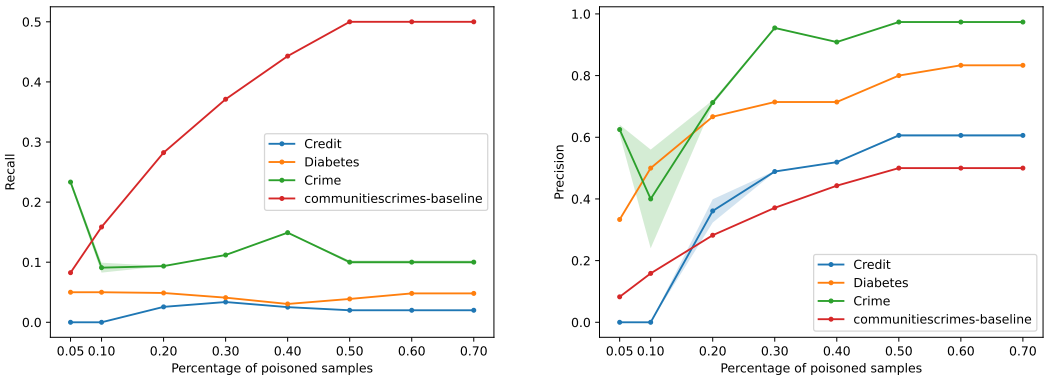
(a) Recall (b) Precision

Figure 30: SVC: Local Outlier Factor method for detection of the poisonous instances in a global poisoning. We report the mean and variance (over all folds) of the recall and precision (larger numbers are better). Note that the variance is not visible because it is close to zero.



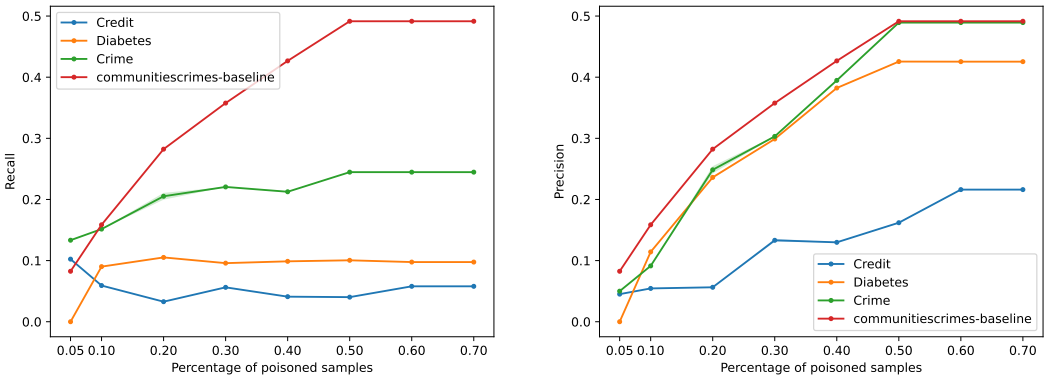
(a) Recall (b) Precision

Figure 31: DNN: Isolation Forest for detection of the poisonous instances in a global poisoning. We report the mean and variance (over all folds) of the recall and precision (larger numbers are better). Note that the variance is not visible because it is close to zero.



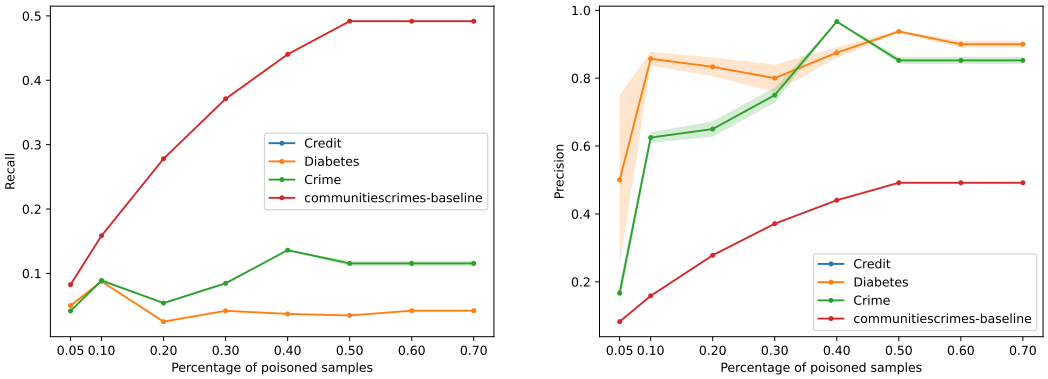
(a) Recall (b) Precision

Figure 32: DNN: Local Outlier Factor method for detection of the poisonous instances in a global poisoning. We report the mean and variance (over all folds) of the recall and precision (larger numbers are better). Note that the variance is not visible because it is close to zero.



(a) Recall (b) Precision

Figure 33: RNF: Isolation Forest for detection of the poisonous instances in a global poisoning. We report the mean and variance (over all folds) of the recall and precision (larger numbers are better). Note that the variance is not visible because it is close to zero.



(a) Recall (b) Precision

Figure 34: RNF: Local Outlier Factor method for detection of the poisonous instances in a global poisoning. We report the mean and variance (over all folds) of the recall and precision (larger numbers are better). Note that the variance is not visible because it is close to zero.