Model Residuals as Shields: A Bilevel Formulation to Defend Smart Grids from Poisoning Attacks

Tung-Wei Lin*, Padmaksha Roy, Yi Zeng,
Ming Jin*, Ruoxi Jia, Chen-Ching Liu, Alberto Sangiovanni-Vinecentelli

***Abstract*—The advancement of interconnected smart grids brings both vast opportunities and heightened cybersecurity risks. Data-driven defense mechanisms, though devised as a shield against these threats, can fall prey to poisoning attacks. We delve into regression settings, underscoring the imperative to fortify defenses against a spectrum of poison ratios, notably those exceeding 0.5—a topic scarcely addressed in prior studies. Recognizing the susceptibilities of smart grids and their manipulable sensors, we exploit the very intent of poisoning attacks—compromising model accuracy—as our defense mechanism. Our proposed bilevel-optimization framework adeptly discerns between poisoned and authentic data based on model residuals, achieving impressive precision and recall. Once sanitized, this model is adaptable for varied applications. Comprehensive evaluations on different smart grid datasets, pitted against myriad poisoning schemes, validate our methodology's edge over existing methods. We also shed light on the implications of model misspecification stemming from temporal autocorrelation, a common feature in smart grid time series.**

***Index Terms*—Adversarial machine learning, poisoning attack, kernel ridge regression, regression, smart grid cybersecurity**

I. Introduction

Smart grids usher in a new era for power systems, integrating renewable energy, optimizing power consumption, and bolstering grid reliability [1], [2]. These grids function as interconnected systems that depend heavily on communication channels for instantaneous data flow among generators, sensors, and controllers [3]. However, this interconnectedness exposes them to potential malicious threats that can compromise their security, resulting in extensive fiscal, infrastructural, and service implications [4]. Robustness in these essential infrastructures is a priority [5].

To identify potential threats, machine-learning (ML) detectors analyze incoming data and command patterns [6]. As an example, linear regression-based detectors at inverters in [7] contrast controller commands with model predictions to block harmful commands. Commands that deviate significantly from predictions are flagged as attacks. Yet, there's a notable weak point in data-driven detectors: they can fall victim to *poisoning attacks*, where adversaries purposefully manipulate training data to influence model outcomes [8], [9]. Defense measures encompass: 1) Input space detection, which leans on robust statistical measures, although often neglecting higher-order nuances [10]. 2) Latent space techniques, specifically designed for neural networks, seeking potential backdoors in the learned space—sometimes through clustering [11]. 3) Prediction signatures, utilizing tools like saliency maps but largely confined to classification tasks such as image recognition [12]. 4) Other methods, including majority voting and differential privacy, but these generally assume a small poisoning ratio [13], [14]. A review of existing literature highlights two primary challenges.

Poisoning attacks in regression settings. While much research centers on classification settings [8], [15], the criticality of regression in smart grids—for tasks like demand prediction and state estimation—presents unique challenges. Unlike classification that targets decision boundaries, poisoning in regression discreetly adjusts predictions, affecting the model's gradient. Such nuanced alterations not only challenge defenses but also escalate risks, since minor deviations in smart grids can result in significant inefficiencies or failures.

Defense against high poison ratios. Diving deeper, an often-overlooked aspect is the criticality of the poison ratio. The majority of studies limit the effectiveness of their methods when the poison ratio is below 0.05 [10] or 0.2 [16], [17]. Yet, our findings indicate a deterioration in performance as this ratio nears 0.5 (refer to Table II). It's acknowledged that M-estimators [18] in robust statistics have an upper limit breakdown point of 0.5 [19]. Although [20] explores high poison ratios in linear regression, it presupposes knowledge of the exact poison ratio and relies on specific distributional assumptions—often unrealistic in practice. Given the diverse data sources in smart grids, from users to smart meters and IoT devices, the potential for external tampering grows. *A high poison ratio (potentially beyond 0.5) prompts a reexamination of current methodologies.* In an environment where the signal masquerades as noise (and vice versa), reliance on conventional techniques becomes questionable. If left unchecked, models derived from such tainted datasets risk significant inaccuracies, potentially jeopardizing grid operations and efficacy.

Contributions & key insights: At their core, poisoning attacks seek to disrupt model accuracy, drastically skewing predictions. This skewing, paradoxically, lights the way to counter these attacks. Generally, a model aligned with normal data will present variances consistent with typical statistical behaviors. However, a poisoned dataset disrupts this behavior, serving as a red flag. Instead of fixating on input features, which can naturally vary, our focus is on the *model's residuals*. Our observations underscore a compelling dichotomy: models optimized for normal data struggle with poisoned datasets and

*Corresponding authors

T.-W. Lin and A. Sangiovanni-Vincentelli are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, California, USA. Email: {twlin, alberto}@eecs.berkeley.edu.

P. Roy, Y. Zeng, M. Jin, R. Jia, and C.C. Liu are with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, Virginia, USA. Email: {padmaksha, yizeng, jinming, ruoxijia, ccliu}@vt.edu.

vice versa, drawing a clear line between them.

Building upon this insight, our methodology tentatively divides data into hypothetical 'normal' and 'poisoned' categories. By training a model to one subset and measuring its error on the other, an iterative refinement of this categorization is possible. This approach is especially adept at navigating high poison ratios in regression contexts. Technically, we employ a bilevel optimization strategy: the inner level focuses on "model training" (e.g., kernel ridge regression), while the outer level seeks to optimize the partition to maximize error differences. We also consider cases where poison ratios approach or exceed 0.5, using the list-decodable setting [20] for evaluation. Our evaluation across diverse smart grid datasets, encompassing various poison ratios and attack methods, indicates that our approach consistently outperforms existing methodologies. Notably, we sidestep the need of predetermined poison ratios, aligning closer with realistic conditions. Compared to state-of-the-art methods such as [16], our method distinguishes between normal and poisoned data with high precision and recall metrics. Beyond data partitioning, the resultant nonlinear model, characterized by its low regression error, is suitable for practical applications. If further model refinement is necessary, the isolated normal dataset provides a reliable foundation.

The remainder of this paper is organized as follows: Sec.II reviews relevant literature on poisoning attacks in smart grid systems and the broader realm of robust machine learning. Our proposed methodology is detailed in Sec.III. Experimental setups and findings are shown in Sec.IV. Limitations of our approach are addressed in Sec.IV-D. The paper concludes in Sec. V.

II. Related Work

Security challenges in smart grids have prompted the development of data-driven detectors against various threats [21]. Traditional techniques like support vector machines target intrusion and eavesdropping attacks, which compromise node access and user privacy respectively [22]. Random forest and neural networks have been explored for jamming attacks that disrupt wireless networks [23]. For electricity theft, wherein customers alter meter readings to reduce bills, studies employ outlier detection [24], [25] and deep learning methods [26]. The data-centric nature of these detectors makes them susceptible to poisoning attacks [27]. Neural networks, although promising [28], might not always be the preferred choice. In contexts where computational and data resources are limited or where model interpretability is crucial, the tilt is towards more transparent models such as linear regression, as seen in distributed energy management systems [7] and smart home power monitoring [29].

Robust linear regression against poisoning attacks has recently garnered attention, yet many works not specifically tailored to smart grid applications cap the poison ratio at a mere 0.05 [10] or 0.2 [16], [17] in their evaluations. Given the open nature of smart grids and potential sensor tampering [28], the training set can exhibit higher poison ratios. For example, smart meter tampering was evaluated with a poison ratio of up to 0.7 in [30]. When the majority of a dataset is poisoned, the list-decodable setting [20] produces multiple functions, one aligning with the ground truth. In our study, we utilize the list-decodable setting for poison ratios of 0.5 and above. Moreover, while the bulk of existing literature assumes data drawn independently from an inherent distribution, we explore model misspecification stemming from autocorrelation in time series data—a prevalent scenario in smart grids.

III. Methodology

Let the dataset be denoted as $\mathcal{D} = \{(x_i, y_i)_{i\in[n]}\}$, where $[n]$ is shorthand for the set $\{1, ..., \}$. The features are represented by $X = \{x_i\}_{i=1}^n \in \mathbb{R}^{n\times d}$ and labels by $Y = \{y_i\}_{i=1}^n \in \mathbb{R}^n$. We consider $\mathcal{H}$ as a Reproducing Kernel Hilbert Space (RKHS) with kernel, $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, such as the polynomial or Radial Basis Function (RBF) kernel [31]. The objective is to identify a function $f : \mathbb{R}^d \to \mathbb{R}$ within $\mathcal{H}$ that yields minimal prediction error for a given feature x amidst the presence of poisons in $\mathcal{D}$.

To crystallize our idea, we define $\mathcal{N}(w) = \{i : w_i = 1\}$ as the index set of hypothetical normal data, with each data point i having a binary weight $w_i \in \{0, 1\}$. Conversely, $\mathcal{P}(w) = \mathcal{N}(1 - w) = \{i : w_i = 0\}$ denotes the index set for the hypothetical poisoned data, being the complement to $\mathcal{N}(w)$. While the hypothesis $w \in \{0, 1\}^n$ delineates the membership sets, the tags of 'normal' or 'poison' merely represent dual data facets; their true nature could be swapped. Within a list-decodable setting, we can train a model using either $\mathcal{N}(w)$ or $\mathcal{P}(w)$ data. Success is declared if one model proves to be accurate. Typically, discerning between the poisoned and normal datasets becomes straightforward upon examination. Now, if we define $\mathcal{L}(f; \mathcal{D}, w) = \frac{1}{\|w\|_1} \sum_{i\in\mathcal{N}(w)} \ell\big(f(x_i), y_i\big)$ as the average error of model f evaluated on hypothetical data defined by w, then $f_w^* = \arg\min_{f\in\mathcal{H}} \mathcal{L}(f; \mathcal{D}, w)$ is the model trained on this hypothetical normal set. We posit that by identifying a hypothesis w, such that the model f_w^*, when trained on the hypothetical normal set, maximizes the error on the hypothetical poisoned set $\mathcal{P}(w)$, it becomes feasible to separate the poisoned data from normal data, resulting in the selection of a dependable model from either f_w^* or f_{1-w}^*.

Building upon the preceding theoretical discussion, we present a bilevel optimization framework as follows:

$$
\begin{aligned}
\max_{\{w_i\}_{i=1}^n} \quad & \frac{1}{\|1-w\|_1} \sum_{i\in\mathcal{P}(w)} \big(y_i - f_w^*(x_i)\big)^2 \\
\text{s.t.} \quad & f_w^* = \arg\min_{f\in\mathcal{H}} \sum_{i\in\mathcal{N}(w)} \big(y_i - f(x_i)\big)^2 + \lambda\|f\|_{\mathcal{H}}^2 \\
& w_i \in \{0, 1\}, \quad \forall i \in [n]
\end{aligned} \tag{1}
$$

Within this framework, the inner level of (1) deduces the best f_w^* from hypothetical normal data, whereas the outer level seeks a w such that f_w^* maximizes the average residuals for the hypothetical poisoned data. Given our focus on regression, we employ squared loss for $\ell(\cdot)$. Moreover, the norm $\|\cdot\|_{\mathcal{H}}$ linked to $\mathcal{H}$, weighted by hyperparameter $\lambda \in \mathbb{R}_+$, serves to regularize smoothness in f. This stems from the rationale that genuine functions tend towards smoothness, in contrast to their aberrant counterparts that may show pronounced fluctuations.

Continuous relaxation of the hypothesis vector. The formulation given by (1) is essentially a mixed-integer quadratic problem, a category which is NP-hard in general [32]. In pursuit of computational tractability, we relax the discrete variable set $\{w_i\}_{i=1}^n$ to be real-valued within the interval $[0,1]$. To provide even more flexibility, we further extend their domain to $\mathbb{R}$ and subsequently apply the Sigmoid function: $s(w_i) = (1+e^{-Tw_i})^{-1}$, where $T \in \mathbb{R}_+$ acts as a temperature parameter, modulating the smoothness of $s(\cdot)$. Accordingly, the related formulation can be written as:

$$\begin{aligned} \max_{\{w_i\}_{i=1}^n} \quad & \frac{1}{\sum_{i=1}^n \big(1-s(w_i)\big)} \sum_{i=1}^n \big(1-s(w_i)\big)\big(y_i - f_w^*(x_i)\big)^2 \\ s.t. \quad & f_w^* = \underset{f\in\mathcal{H}}{\arg\min} \sum_{i=1}^n s(w_i)\big(y_i - f(x_i)\big)^2 + \lambda\|f\|_{\mathcal{H}}^2 \\ & w_i \in \mathbb{R}, \quad \forall i \in [n] \end{aligned} \tag{2}$$

Reduction to single-level optimization. Despite the continuous relaxation above, the computational complexity of a bilevel optimization remains. Nevertheless, by invoking the representer theorem [33], we can derive the unique optimal solution for the inner level. Specifically, this solution can be expressed as: $f(x) = \sum_{i=1}^n \alpha_i k(x, x_i)$, where $\alpha = \{\alpha_i\}_{i=1}^n \in \mathbb{R}^n$. Substituting this expression into the inner level of (2) to determine the optimal α^* yields: $\alpha^* = \big(S(w)K + \lambda I\big)^{-1} S(w)Y$, where $S(w)$ is a diagonal matrix with $S(w)_{i,i} = s(w_i)$. Additionally, $K \in \mathbb{R}^{n\times n}$ is the kernel matrix defined such that $K_{i,j} = k(x_i, x_j)$, and I is the identity matrix. A detailed derivation can be found in App. B.

Consequently, our bilevel optimization problem can be reduced to a single-level optimization. By substituting $f_w^* = \sum_{i=1}^n \alpha_i^* k(x, x_i)$ into the outer optimization, we can express the problem in a vectorized form as:

$$\max_{w\in\mathbb{R}^n} \quad H(w) := \frac{1}{\mathrm{Tr}\big(I - S(w)\big)} \Big(\big(I - S(w)\big)^{\frac{1}{2}}(Y - \Lambda)\Big)^2, \tag{3}$$

where $\Lambda := K\big(S(w)K + \lambda I\big)^{-1} S(w)Y$ and $\mathrm{Tr}(\cdot)$ denotes the trace operation.

The problem posed by (3) is a non-concave maximization problem, lacking discernible structure. Consequently, we employ gradient ascent to solve it. The gradient of $H(w)$ with respect to w_i is analytically derived as:

$$\begin{aligned} \frac{\partial H(w)}{\partial w_i} \quad &= \frac{s'(w_i)}{\Big(\mathrm{Tr}\big(I - S(w)\big)\Big)^2} \Big(\big(I - S(w)\big)^{\frac{1}{2}}(Y - \Lambda)\Big)^2 \\ &\quad + \frac{1}{\mathrm{Tr}\big(I - S(w)\big)} \Big(s'(w_i)(-y_i^2 + 2\Lambda_i y_i - \Lambda_i^2) \\ &\quad + \Big(2(\Lambda - Y)^\top \big(I - S(w)\big)\Big) \tfrac{\partial \Lambda}{\partial w_i}\Big), \end{aligned} \tag{4}$$

where $\frac{\partial \Lambda}{\partial w_i} = K\big(S(w)K + \lambda I\big)^{-1} e_i s'(w_i)\big(e_i^\top (Y - \Lambda)\big)$, e_i is the canonical unit vector with the i-th entry being 1 and others being 0, and $s'(w_i)$ is the derivative of $s(w_i)$. A detailed derivation is in App. C.

Scalability via Random Fourier Features (RFF). To evaluate (4), we must compute both Λ and $\frac{\partial \Lambda}{\partial w_i}$, which entails the inversion of an $n \times n$ matrix $\big(S(w)K + \lambda I\big)^{-1}$. The complexity of this inversion increases with the dataset's size, presenting a computational bottleneck intrinsic to kernel methods. To address this, we utilize the RFF approximation [34], which estimates shift-invariant kernels, encompassing the widely-used RBF kernel defined by $k(x_i, x_j) = \exp(-\gamma\|x_i - x_j\|^2)$, where $\|\cdot\|$ denotes the Euclidean distance.

In the RFF approach, the Fourier transform of the kernel is sampled randomly p times, and this sampling transforms the raw data x_i into a p-dimensional space. A prevalent transformation is:

$$z(x_i) = \sqrt{\frac{2}{p}} \big[\cos(\omega_1^\top x_i + b_1), \cdots, \cos(\omega_p^\top x_i + b_p)\big]^\top,$$

where $\omega_i \sim q(\omega), b_i \sim \mathrm{Uniform}(0, 2\pi)$, and $q(\omega)$ is the Fourier transform of the kernel. Leveraging Bochner's Theorem [35] and defining Z as the $n \times p$ matrix with its i-th row given by $z(x_i)^\top$, we can approximate K by $ZZ^\top$ (refer to [34] for a comprehensive exposition).

With RFF approximation and the Woodbury identity [36], we can sidestep the necessity of inverting an $n \times n$ matrix. To elucidate:

$$\begin{aligned} \Lambda \quad &= K(S(w)K + \lambda I)^{-1} S(w)Y \\ &\approx ZZ^\top (S(w)ZZ^\top + \lambda I)^{-1} S(w)Y \\ &= Z(Z^\top S(w)Z + \lambda I)^{-1} Z^\top S(w)Y \\ &:= Z\theta. \end{aligned} \tag{5}$$

This approach grants us the flexibility to manage computational complexity directly since the operation now revolves around inverting a $p \times p$ matrix. A larger p enhances approximation accuracy, but concurrently increases computational cost.

For kernels that are not shift-invariant, such as the linear kernel ($k(x_i, x_j) = x_i^\top x_j$) or the polynomial kernel ($k(x_i, x_j) = (x_i^\top x_j + c)^m$, with $m < n$), the $n \times n$ matrix inversion can also be skirted using the Woodbury identity, without the explicit computation of the kernel matrix K. Using the linear kernel as an illustrative example, where $K = XX^\top$:

$$\begin{aligned} \Lambda \quad &= K(S(w)K + \lambda I)^{-1} S(w)Y \\ &= XX^\top (S(w)XX^\top + \lambda I)^{-1} S(w)Y \\ &= X(X^\top S(w)X + \lambda I)^{-1} X^\top S(w)Y. \end{aligned} \tag{6}$$

Algorithm. The outlined procedure is detailed in Alg. 1. Upon convergence of the gradient ascent, $s(w_i)$ serves as an indicator of the contribution of the i-th data point towards the learning of f_w^*. Specifically, when $s(w_i)$ approaches 1, it implies that the data point (x_i, y_i) has notably influenced the learning of f_w^*. Given that the choice to use a data point for f_w^* is inherently binary, we round $s(w_i)$ to the nearest integer, either 0 or 1, and denote the result as $\bar{w}_i$. This step enables us to discern and segregate the learned normal dataset, $\mathcal{N}(\bar{w})$, from the complementary set of anomalous or poisoned data, $\mathcal{P}(\bar{w})$.

IV. Experiments

Datasets. The Stability dataset [37] examines a decentralized four-node electrical system's local stability, focusing on 11 features such as participant reaction time and nominal power consumption. The goal is to predict the maximal real

Algorithm 1 Gradient Ascent

```
Inputs: data $X, Y$, regularization hyperparameter $\lambda$, temperture parameter $T$, number of samples for RFF approximation $p$, kernel function $k$, number of iterations for gradient ascent $t$, learning rate $\beta$
Output: $\mathcal{N}(\bar{w}), \mathcal{P}(\bar{w}), f^*_{\bar{w}}, f^*_{1-\bar{w}}$
Calculate the Fourier transform $q$ of $k$
Draw $p$ i.i.d. samples $\omega_1, \dots, \omega_p$ from $q$ and $b_1, \dots, b_p$ from Uniform$(0, 2\pi)$
Construct $Z$, whose $i$-th row is $z(x_i) = \sqrt{\frac{2}{p}}[cos(\omega_1^\top x_i + b_1), \cdots, cos(\omega_p^\top x_i + b_p)]$
Initialize $w^{(1)} = 0$
for $j = 1, 2, \dots, t$ do
    $w^{(j+1)} = w^{(j)} + \beta \nabla_w H(w^{(j)})$
    // $\nabla_w H(w^{(j)})$ is calculated according to (4)
end for
$\{\bar{w}_i\}_{i=1}^n \leftarrow$ Round $\{s(w_i)\}_{i=1}^n$ to the nearest integer
$\mathcal{N}(\bar{w}) = \{i : \bar{w} = 1\}, \mathcal{P}(\bar{w}) = \{i : \bar{w} = 0\}$
$f^*_{\bar{w}}(x) = z(x)^\top (Z^\top diag(\bar{w}) Z + \lambda I)^{-1} Z^\top diag(\bar{w}) Y$
$f^*_{1-\bar{w}}(x) = z(x)^\top (Z^\top diag(1-\bar{w}) Z + \lambda I)^{-1} Z^\top diag(1-\bar{w}) Y$
return $\mathcal{N}(\bar{w}), \mathcal{P}(\bar{w}), f^*_{\bar{w}}, f^*_{1-\bar{w}}$
```

part of a characteristic equation root, where positive indicates instability and negative denotes stability. The DERMS dataset [7] revolves around a smart grid's Distributed Energy Resource Management System (DERMS). It logs measurements sent to and actions from the DERMS controller, covering features including real and reactive powers, and each node's maximum inverter generation limit. Predictions from this dataset assist in detecting deviations in control actions, signaling potential intrusions. The House dataset [38] describes a low-energy home's electricity usage through 28 features like kitchen humidity and local weather data. It forecasts the household's total appliance energy consumption, aiding in anomaly detection or demand management. Lastly, the CCPP dataset [39] delves into a Combined Cycle Power Plant, capturing 4 features such as ambient temperature/humidity and exhaust vacuum. The objective is to predict the plant's net hourly energy output.

Poisoning Attacks. We outline the data poisoning attacks used in our experiments. Given a normal dataset $\mathcal{D}_n$, their goal is to construct the resultant dataset $\mathcal{D} = (\mathcal{D}_n \setminus \mathcal{D}_t) \cup \mathcal{D}_p, \mathcal{D}_t \subset \mathcal{D}_n$, with $\mathcal{D}_t$ representing tampered data from $\mathcal{D}_n$ and $\mathcal{D}_p$ as the injected poisons. The *poison ratio* is given by $\frac{|\mathcal{D}_p|}{|\mathcal{D}|}$.

In the Optimization-Based Injection [16], a bilevel optimization framework crafts $\mathcal{D}_p$. The attacker, knowing dataset $\mathcal{D}'$ (similar to $\mathcal{D}_n$) and loss function $\ell(\cdot)$, maximizes the average loss on a validation set using an optimal model from the inner level. This model represents the impact of injecting $\mathcal{D}_p$ into $\mathcal{D}_n$. In tests, we set $\mathcal{D}' = \mathcal{D}_n$, implying full attacker knowledge. The Flip method [17] also uses a similar dataset, but without requiring access to the loss function. Poisons are crafted based on a feasibility domain of the labels to avert suspicion, and each poison corresponds to data points in $\mathcal{D}'$. Like before, $\mathcal{D}' = \mathcal{D}_n$ in our tests, which allows us to consider a poison ratio of 0.5. In the Additive Attack [30], labels in $\mathcal{D}_n$ are modified by adding a fixed value, $\delta_a = 2$ in our tests. This misleads the model to predict higher values. Lastly, in Noise Corruption, labels in $\mathcal{D}_n$ are altered using values from a distribution, specifically $\delta_n \sim N(0, 0.2)$ in our tests, reflecting data acquisition noise.

The first two attacks maintain $\mathcal{D}_n$ while introducing extra poisons ($\mathcal{D}_t = \{\phi\}$), which require database access. However, such attacks are not always possible, such as the case when the number of data entries is pre-set. On the other hand, the last two attacks directly change the labels in $\mathcal{D}_n$ ($\mathcal{D}_t \neq \{\phi\}$), which can happen during data acquisition. For instance, during man-in-the-middle attacks, the attacker may swap poisons for standard data, retaining dataset size post-assault.

Baseline Defenses. TRIM [16] iteratively refines the function f by focusing on data subsets with the smallest residuals, predicated on a given poison ratio. Its successor, iTRIM [17], enhances this by repeatedly applying TRIM across different poison ratios to accommodate situations when the precise ratio is unknown. Huber loss [18], an M-estimator offshoot, combines the sum of squares loss and sum of absolute values loss. Recognizing the former's vulnerability to outliers, it reduces the weights for penalties when encountering residuals that may signify poisons. Lastly, Sever [10], assuming the true poison ratio, pinpoints poisons by evaluating their gradient characteristics. This method filters out data points with notably strong projected gradients on the leading singular vector of the gradient matrix.

Setup. The proposed method is implemented in Python while the baseline defenses are from the sklearn package [40] or the authors [10], [17] respectively. iTRIM is included instead of TRIM because the former does not assume knowledge of the poison ratio. For each dataset, we randomly select $8,000$ to $10,000$ data points as $\mathcal{D}_n$ and use the rest as the test set. Both the features and labels are scaled to the range $[0, 1]$. The regularization hyperparameter λ is set to 0.01, the temperature T to 0.5, the number of draws for RFF p to 200, the number of iterations to 3500, the learning rate β to 0.01, and the RBF kernel as k.

Evaluation Metrics. To evaluate the partition of $\mathcal{D}$ into poisons and normal data, we define poisons as positives, normal data as negatives, and calculate the precision and recall. Denote true positives, false positives, true negatives, and false negatives as TP, FP, TN, FN, respectively. Precision and recall can be calculated as follows

$$\text{precision} = \frac{\text{TP}}{\text{TP+FP}}, \quad \text{recall} = \frac{\text{TP}}{\text{TP+FN}}. \tag{7}$$

Moreover, we evaluate the model learned from the partitioned normal data on the test set using mean squared error (MSE). In the following, we refer to MSE as the test MSE unless stated otherwise.

A. Poison Ratio Smaller Than 0.5

In this section, we vary the poison ratio from 0 to 0.4.

Does the proposed method partition data more effectively? High precision signifies a reduced chance of erroneously classifying normal data as poisoned, while high recall

TABLE I: Precision / Recall (%) at poison ratios smaller than 0.5

	Stability								DERMS							
	Flip		Opt		Add		Noise		Flip		Opt		Add		Noise	
ratio	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM
0.1	**100**/**100**	93.2/**100**	**99.9**/**100**	93.2/**100**	**99.9**/**100**	93.2/**100**	35.2/**67.8**	**83.8**/32.3	**85.5**/**96.7**	63.3/96.6	**70.4**/**95.6**	62.6/**95.6**	**100**/**100**	65.5/**100**	14.1/6.4	**34.3**/**31.3**
0.2	**100**/**100**	96.9/**100**	**100**/**100**	96.9/**100**	**100**/**100**	96.9/**100**	60.4/**60.4**	**94.0**/26.6	**98.0**/**97.9**	80.8/**97.9**	**92.0**/**96.0**	78.9/95.7	**100**/**100**	82.5/**100**	24.8/5.7	**52.3**/**28.0**
0.3	**100**/**100**	98.2/**100**	**100**/**100**	98.2/**100**	**100**/**100**	98.2/**100**	78.2/**54.7**	**97.3**/24.0	**96.4**/**98.1**	88.3/**98.1**	**95.3**/90.6	88.2/**95.4**	**100**/99.9	87.7/**100**	34.8/5.5	**64.1**/**26.3**
0.4	**100**/**100**	10.0/7.4	**100**/99.9	98.8/**100**	**100**/**100**	98.8/**100**	87.3/**49.7**	**98.6**/22.4	97.8/**97.8**	**99.6**/18.5	**97.5**/94.0	92.7/**95.4**	**100**/**100**	92.6/**100**	45.9/5.5	**73.8**/**25.5**
	House								CCPP							
	Flip		Opt		Add		Noise		Flip		Opt		Add		Noise	
ratio	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM
0.1	**98.7**/**99.9**	72.4/**99.9**	**90.9**/**98.8**	80.3/**98.7**	**99.9**/**100**	81.4/**100**	42.4/1.4	**46.7**/**26.4**	**99.1**/**100**	93.2/**100**	**98.0**/**100**	93.2/**100**	**100**/**100**	93.2/**100**	90.9/**33.3**	**93.2**/18.4
0.2	**99.2**/**100**	91.4/**100**	**98.0**/**99.2**	90.6/**99.2**	**100**/**100**	91.4/**100**	65.5/1.0	**67.0**/**24.9**	**99.8**/**100**	96.9/**100**	**99.7**/99.9	96.9/**100**	**100**/**100**	96.9/**100**	97.0/25.2	**97.1**/**27.5**
0.3	**99.6**/**100**	95.2/**100**	**99.8**/**99.2**	94.4/**99.2**	**100**/**100**	95.2/**100**	70.4/0.7	**76.2**/**23.1**	**99.9**/99.9	98.2/**100**	**99.8**/**100**	98.1/**100**	**100**/**100**	98.2/**100**	93.2/**53.1**	**98.2**/24.3
0.4	99.7/**99.1**	**99.8**/89.5	**100**/**99.2**	96.3/**99.2**	**100**/**100**	97.1/**100**	80.8/0.5	**84.2**/**22.6**	**100**/**99.9**	99.2/84.9	**99.9**/97.7	**99.9**/**99.3**	**100**/**100**	98.8/**100**	97.8/**43.9**	**98.7**/22.5

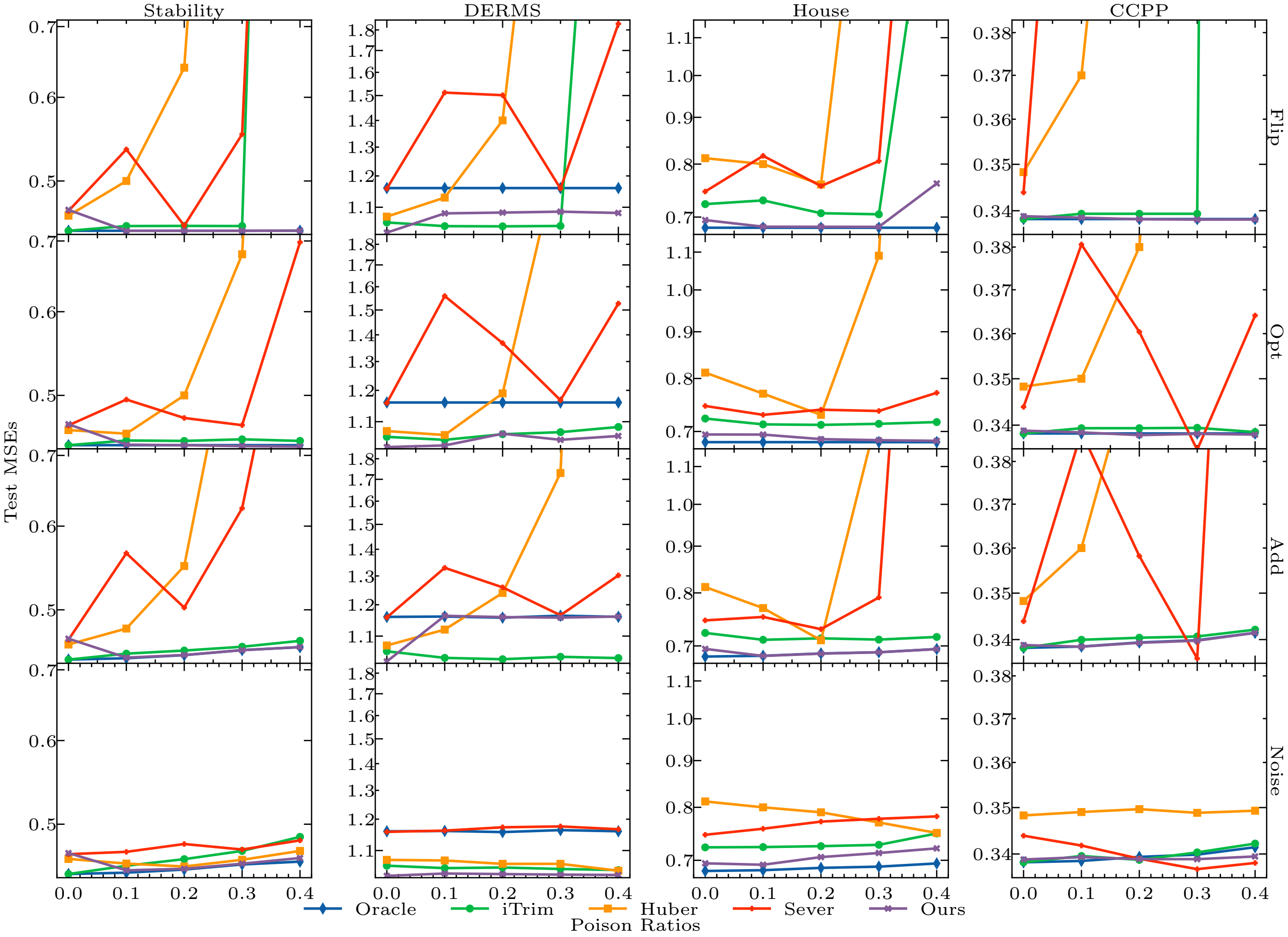

Fig. 1: **Test MSEs at poison ratios smaller than 0.5.** A partition with higher precision and recall leads to lower test MSE if little or no model misspecification exists. Lower test MSEs by iTRIM on DERMS are achieved by sacrificing precision because of model misspecification.

indicates a reduced chance of mistakenly identifying poisoned data as normal. Both these measures are indicative of effective data partitioning. In Table I, we juxtapose the precision and recall of our approach with that of iTRIM. For the additive attack, our method consistently delivers nearly 100% precision and recall across various datasets and poison ratios. We also observe substantially higher precision and recall for Flip and Optimization-based attacks using our approach. For noise corruption scenarios, the precision and recall metrics for both methods drop in comparison to other attack types. This decline can be attributed to the fact that the noise, δ_n, is sampled from a distribution $N(0, 0.2)$. Consequently, if a sampled noise has a small magnitude, it does not significantly impact the poisoning effect on f.

Does a better partition lead to a better model? We show the test MSEs of the model learned from the oracle and baseline methods in Fig. 1. The oracle is the data partition with 100% precision and recall. In the following, we discuss the case when the poison ratio is above and at zero separately.

At poison ratios above zero, our method has the smallest MSEs compared with the baselines except for under Flip and additive attack on DERMS. The reason is that DERMS is heavily autocorrelated (See App. A for the autocorrelation profile of each dataset.). This model misspecification brings about the fact that higher precision and recall don't necessarily lead to lower MSE of the resultant model. iTRIM achieves

TABLE II: Test MSEs at 0.5 poison ratio

	Stability							DERMS							House							CCPP						
	$\lambda = 0.01$				λ	Ours	Oracle MSE	$\lambda = 0.01$				λ	Ours	Oracle MSE	$\lambda = 0.01$				λ	Ours	Oracle MSE	$\lambda = 0.01$				λ	Ours	Oracle MSE
	iTRIM	Huber	Sever	Ours				iTRIM	Huber	Sever	Ours				iTRIM	Huber	Sever	Ours				iTRIM	Huber	Sever	Ours			
Flip	28.48	9.39	23.50	3.16	10	**0.79**	0.45	51.39	18.20	65.89	25.01	50	**1.44**	1.16	29.35	21.35	58.46	8.78	100	**0.69**	0.68	32.31	12.10	29.45	0.56	50	**0.34**	0.34
Opt	27.41	10.44	23.33	1.83	0.1	**1.54**	0.45	52.17	18.92	62.74	2.39	100	**1.62**	1.16	47.97	21.41	44.46	22.51	0.1	**1.32**	0.68	43.93	12.74	37.66	0.50	0.1	**0.43**	0.34
Add	148.2	102.9	0.49	127.2	100	**0.47**	0.46	251.0	99.07	158.0	7.89	100	**1.16**	1.15	208.3	103.0	179.3	139.0	100	**0.69**	0.69	330.1	100.8	171.9	56.6	100	**0.34**	0.34
Noise	0.50	**0.47**	180	0.50	0.1	**0.47**	0.46	**1.04**	1.05	1.16	1.06	0.1	**1.04**	1.15	0.75	0.74	0.79	0.76	0.1	**0.73**	0.69	**0.34**	0.35	**0.34**	**0.34**	0.1	**0.34**	0.34

TABLE III: False positives and test MSEs of each method at 0 poison ratio

	iTRIM		Sever		Ours		Oracle	Training Set
	FP	MSE	FP	MSE	FP	MSE	MSE	Size
Stability	0	**0.449**	6072	0.468	253	0.470	**0.449**	8000
DERMS	652	1.055	6680	1.159	379	**1.025**	1.160	8800
House	377	0.723	7442	0.747	101	0.695	**0.681**	9800
CCPP	0	**0.338**	6831	0.344	31	0.339	**0.338**	9000

lower MSEs by sacrificing precisions as shown in Table I. By using kernel ridge regression, we assume the data follow $y = z(x)^\top \theta + \epsilon$, where ϵ denotes the error that includes model misspecification error and i.i.d. noise. According to the Gauss-Markov theorem [41], linear regression is only the best linear unbiased estimator (BLUE) when ϵ is uncorrelated; otherwise, it leads to inflated MSEs. Therefore, by removing normal data with high model misspecification errors, iTRIM achieves lower MSEs. When the sifted normal data is used for fitting sophisticated downstream models, a lower precision may lead to degraded performance.

Note that for a fair comparison, we leave the poison ratio parameter in Sever [10] as default in its original implementation, which is 0.3. Hence, a drop in MSE is observed at 0.3 in several cases.

At a poison ratio of 0, corresponding to scenarios where no poisoning attack has occurred, the FPs and the associated MSEs are detailed in Table III. Huber is omitted from this discussion as it doesn't have the capability to filter out poisons. Without model misspecification, a lower FP count typically results in reduced MSEs. This relationship is evident in the cases of Stability and CCPP, where iTRIM reports 0 FP. However, in scenarios exhibiting significant autocorrelation, such as DERMS, having non-zero FPs can yield even lower MSEs than the oracle. This behavior sharply contrasts with the instances where DERMS registers a high precision but elevated MSE when the poison ratio exceeds 0. One potential reason for this observed behavior is that the outer level of equation (2) can result in a 'divide-by-zero' scenario if $w_i = 1$ for all $i \in [n]$. Consequently, at a poison ratio of 0, our methodology may incline towards omitting normal data points that exhibit pronounced model misspecification errors. This tendency, in turn, results in non-zero FPs but minimized MSEs.

B. Poison Ratio at 0.5

Is our method directly applicable to the list-decodable setting? When the poison ratio is exactly at 0.5, an inductive bias is necessary because there are an equal number of poisons and normal data. Our insight is that effective poisons are inevitably far from the normal data. Therefore, smoothness should be encouraged to prevent learning from portions of poisons and normal data.

We visualize the importance of inductive bias with a 1-dimensional synthetic dataset generated by drawing 10,000 random samples in the range $x \in [-3, 2]$ from the noisy function $y = x^2 + sin(x^3) + exp(x+1) - 2 + \xi$, where $\xi \sim N(0, 0.5)$. Both x and y are then scaled to $[0, 1]$. Among the 10,000 data points, 8000 are used as $\mathcal{D}_n$ and the rest as the test set. $\mathcal{D}_n$ and the true function f_{true} are visualized in Fig. 2(a). Next, Flip creates $\mathcal{D}_p$ of size 8000, resulting in a 0.5 poison ratio in Fig. 2(b). We also show the oracle model f_o with $\lambda = 0.01$, whose test MSE is 0.085.

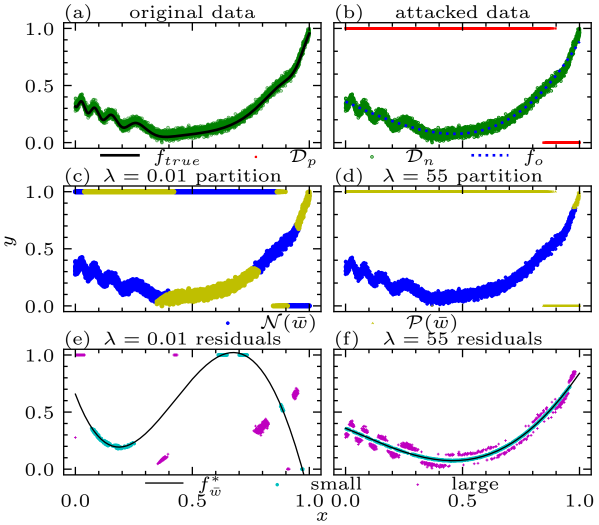

Fig. 2: **Visualization of synthetic data to illustrate the importance of inductive bias at 0.5 poison ratio.** When λ is increased from 0.01 to 55 to encourage smoothness, the partition becomes more ideal and the resultant model captures the true function better.

Fig. 2(c) visualizes $\mathcal{N}(\bar{w})$ and $\mathcal{P}(\bar{w})$ that our proposed method produces when $\lambda = 0.01$. An ideal partition in the list-decodable setting is either of the following cases: (i) $w_i \in \mathcal{N}(\bar{w})$ iff $w_i \in (\mathcal{D}_n \backslash \mathcal{D}_t)$, or (ii) $w_i \in \mathcal{P}(\bar{w})$ iff $w_i \in (\mathcal{D}_n \backslash \mathcal{D}_t)$. Nevertheless, in Fig. 2(c), both the poisons and the normal data are divided into sections by the partition. We further sample along the x axis and plot $f^*_{\bar{w}}(z(x))$ in Fig. 2(e), along with 1000 data points in $\mathcal{N}(\bar{w})$ with the largest and smallest residuals respectively. As illustrated, $f^*_{\bar{w}}$ with $\lambda = 0.01$ is a lot more wiggly than f_{true} because a highly wiggly function at 0.5 poison ratio that oscillates between poisons and normal data maximizes the outer objective function in (2). The test MSE in this case is 27.79, which is highly influenced by the poisons. Next, we increase λ to encourage a smooth $f^*_{\bar{w}}$. In Fig. 2(d), we set $\lambda = 55$ on line 6 in Alg. 1 to produce a more ideal partition. Subsequently, $f^*_{\bar{w}}$ is learned from $\mathcal{N}(\bar{w})$, setting $\lambda = 0.01$ on line 10 in Alg. 1 to compare with the test MSE from f_o. The test MSE is greatly reduced from 27.79 to 0.093 and the resultant $f^*_{\bar{w}}$ captures the true function much better as shown in Fig. 2(f).

TABLE IV: Precision / Recall (%) at poison ratios larger than 0.5

	Stability						DERMS					
	Opt		Add		Noise		Opt		Add		Noise	
ratio	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM
0.6	71.2/**96.0**	**74.2**/73.6	**100/100**	**100**/99.2	96.5/**42.1**	**99.2**/20.3	92.2/**96.8**	**96.4**/94.5	**100/100**	**100**/94.7	66.4/6.0	**86.0/23.9**
0.7	74.5/**99.5**	**82.3**/73.5	**100/100**	**100**/99.2	98.1/**38.8**	**99.5**/19.6	95.8/**95.9**	**98.1**/94.6	**100/100**	**100**/95.2	75.3/6.0	**90.0/23.2**
0.8	82.2/**99.9**	**89.3**/74.4	**100/100**	**100**/99.2	99.0/**36.3**	**99.7**/19.1	95.7/**95.4**	**99.5**/94.2	**100/100**	**100**/94.7	85.9/6.2	**94.5/22.9**
0.9	91.5/**99.4**	**95.2**/71.4	**100/100**	**100**/99.2	99.6/**33.7**	**99.9**/18.5	97.0/**95.3**	**99.9**/94.0	**100/100**	**100**/94.2	93.2/5.8	**97.2/22.3**
	House						CCPP					
	Opt		Add		Noise		Opt		Add		Noise	
ratio	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM	Ours	iTRIM
0.6	98.7/**99.2**	**100/99.2**	99.6/**100**	**100**/98.0	**93.0/40.6**	92.8/21.3	79.3/**99.4**	**92.3**/90.5	**100/100**	**100**/99.2	98.9/**42.8**	**99.5**/20.4
0.7	98.9/**99.3**	**100**/99.2	**100/100**	**100**/97.8	95.2/**37.9**	**95.6**/19.6	83.9/**99.5**	**95.6**/91.0	**100/100**	**100**/99.2	99.4/**39.7**	**99.7**/19.6
0.8	99.6/**99.3**	**100**/99.2	**100/100**	**100**/97.7	97.2/**35.0**	**97.6**/20.3	88.9/**99.3**	**97.4**/90.8	**100/100**	**100**/99.2	99.7/**36.6**	**99.8**/19.0
0.9	99.9/**99.2**	**100/99.2**	**100/99.9**	**100**/95.8	**98.8/32.7**	98.8/18.3	94.5/**98.9**	**99.0**/90.2	**100/100**	**100**/99.2	99.9/**34.1**	**99.9**/18.5

TABLE V: λ configurations at poison ratios larger than 0.5

	Stability			DERMS			House			CCPP		
	Ours	iTRIM	Sever	Ours	iTRIM	Sever	Ours	iTRIM	Sever	Ours	iTRIM	Sever
Opt	10^{-3}	10^{-2}	10^{-3}	10^{-3}	10^{-2}	10^{-3}	10^{-2}	10^{-2}	10^{-2}	10^{-2}	10^{-2}	10^{-1}
Add	10^{-2}	10^{-3}	10^{-2}	10^{-2}	10^{-2}	10^{-3}	10^{-2}	10^{-3}	10^{-2}	10^{-2}	10^{-3}	10^{-1}
Noise	10^{-1}	10^{-2}	10^{-1}	10^{-1}	10^{-2}	10^{-1}	10^{-2}	10^{-2}	10^{-1}	10^{-3}	10^{-2}	10^{-2}

Does the insight transfer to high-dimensional data? We attack the datasets at 0.5 poison ratio and compare the test MSEs of different defense methods. First, λ is set to 0.01 and the corresponding MSEs are presented in Table II. To discern which of $f^*_{\bar{w}}$ and $f^*_{1-\bar{w}}$ learns the normal data, a small set of 10 normal data points is partitioned from $\mathcal{D}_n$ to form a validation set. We only include the test MSE of $f^*_{\bar{w}}$ or $f^*_{1-\bar{w}}$, whichever has the smaller validation MSE. The results show that without an inductive bias, at $\lambda = 0.01$, all attacks increase the MSEs by a large margin, with one exception being noise corruption. The reason is similar to as mentioned in Sec. IV-A. Some δ_n may have small magnitudes, resulting in ineffective poisons.

Then, inductive bias is introduced by increasing λ on line 6 in Alg. 1. Specifically, we choose λ among $\{0.1, 10, 50, 100\}$ and select the one that has the lowest validation MSE at $\lambda = 0.01$. Note that λ on line 10 remains 0.01. As Table II shows, our test MSEs decrease to a more tolerable level for all cases. Results from other methods under different λ's are omitted because they either are insensitive to λ or produce unsatisfactory results compared with ours. The reason iTRIM yields unsatisfactory performance is that it initializes by learning f on the entire $\mathcal{D}$. This results in large residuals on both poisons and normal data at 0.5 poison ratio, which impedes distinguishing poisons and normal data via residuals.

C. Poison Ratio Larger Than 0.5

Finally, in this section, we discuss when the poison ratio is larger than 0.5. As in Sec. IV-B, the list-decodable setting is adopted, and the final output function is chosen as the one with the smaller validation MSE. Moreover, since over half of the data are poisons, $f^*_{\bar{w}}$ in (2) tends to fit the poisons instead of the normal data. We thus use the validation set to also determine the appropriate λ to set on line 6 in Alg. 1. Table V shows the configuration for each dataset and attack. We do not include Flip because it supports at most 0.5 poison ratio. The same list-decodable setting and search for appropriate λ are applied to iTRIM and Sever. However, $\lambda = 0.01$ for all of Huber's experiments since it does not filter poisons out.

Does our method still yield better partitions at poison ratios larger than 0.5? We show the precision and recall of our method and iTRIM in Table IV. In several cases other than additive attack, where we have nearly perfect precision and recall, iTRIM yields higher precision but lower recall than our method. iTRIM's behavior of mistakenly excluding data in learning f was also observed in Table III, where iTRIM has generally higher recall than precision. Now that f learns the distribution of the poisons instead of the normal data, the poisons not captured by f lead to lower recalls. On the other hand, our method is able to include more poisons in learning f but at the same time mistakenly includes more normal data in learning f, which leads to lower precisions.

How do precision and recall affect resultant models in the presence and absence of model misspecification? We show the test MSEs in Fig. 3. It appears that when model misspecification exists, the MSEs benefit from iTRIM's method. Specifically, iTRIM has lower MSE on DERMS and House under optimization-based injection. However, our method achieves lower MSEs on Stability and CCPP. Nearly perfect precision and recall under additive attack also give us the lowest MSE across all datasets. As for noise corruption, Huber achieves the lowest MSE on House, iTRIM on DERMs, and our method on the datasets where there is no model misspecification. Furthermore, since optimization-based injection creates poisons in addition to the normal data, $\mathcal{D}$ becomes $10\times$ larger at 0.9 poison ratio. As a result, Sever runs out of memory on House at poison ratio 0.8 and all datasets at poison ratio 0.9.

D. Discussion and Limitations

In this section, we explore a potential vulnerability inherent to our formulation. The underlying assumption of our approach posits that normal data and poisons distinctly segregate into two distributionally separate clusters within the training dataset. Nevertheless, this presupposition falters when the training set encompasses multiple poison distributions. This challenge is not exclusive to our method; many defensive strategies grapple with this very issue, as elucidated in [42]. However, in scenarios where the poison ratio is relatively low, or when the poison distribution closest to the regular data remains sufficiently distinct, our technique consistently and robustly differentiates poisons from authentic data points. A more comprehensive experimental demonstration can be found in Appx. D.

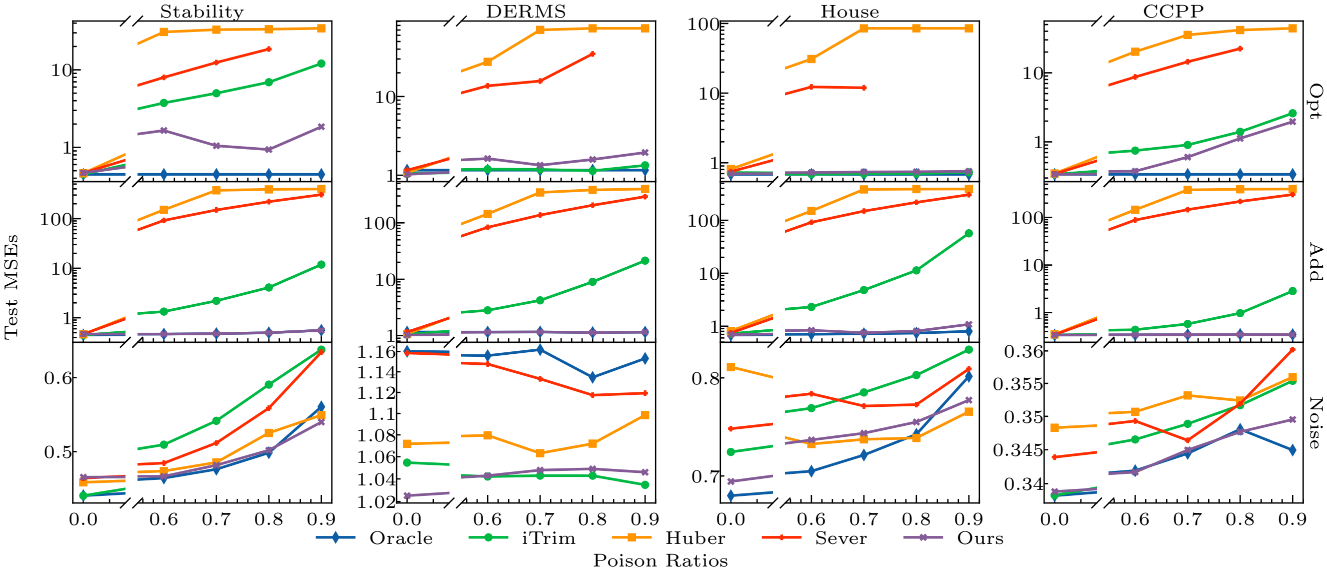

Fig. 3: **Test MSEs at poison ratios larger than 0.5.** According to Table IV, nearly 100% precision and recall on additive attack lead to the lowest test MSE. High recall benefits datasets without model misspecification (Stability and CCPP), while high precision benefits datasets with model misspecification (DERMS and House).

V. Conclusion

In this research, we presented a bilevel optimization-based framework to counteract poisoning attacks targeting data-driven smart grid applications. Notably, our approach achieves high precision and recall in partitions for poison ratios below 0.5, culminating in models with the lowest test MSE, especially when no model misspecification exists. With an incorporated smoothness inductive bias, our method excels, registering the lowest test MSE at a 0.5 poison ratio. For elevated poison ratios, our method prioritizes recall, still benefiting the subsequent model in scenarios without misspecification. We further explored the impact of model inaccuracies due to autocorrelation across varied poison ratios, as well as a potential vulnerability in our approach. Looking forward, adapting the bilevel optimization for autoregressive models stands as a promising avenue to eliminate model discrepancies.

Appendix A
Autocorrelation in the Datasets

For each dataset, we plot the autocorrelation function of the training residuals after fitting a kernel ridge regression model up until lag 50 in Fig. 4. The figure shows that Stability and CCPP are free of autocorrelation while House and DERMS are autocorrelated. In particular, DERMS is more heavily autocorrelated than House. This indicates the existence of model misspecification, where our kernel ridge regression does not capture the interaction between data points of different time stamps. In fact, any non-autoregressive model fails to capture autocorrelation.

Appendix B
Derivation of α^*

Given the following optimization problem.

$$f^* = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} s(w_i)\big(y_i - f(x_i)\big)^2 + \lambda |f|_{\mathcal{H}}^2 \tag{8}$$

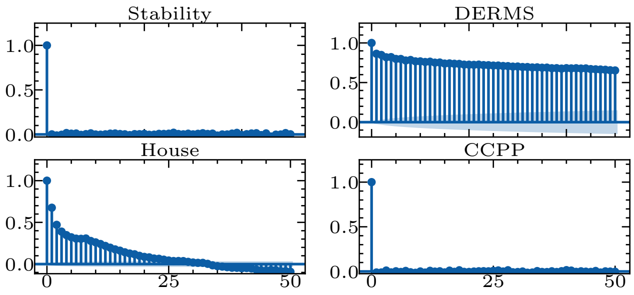

Fig. 4: **Autocorrelation function of the residuals of each dataset.**

Due to representer theorem $f^*(x) = \sum_{i=1}^{n} \alpha_i^* k(x, x_i)$,

$$\begin{aligned} f^* \quad &= \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} s(w_i)\Big(y_i - \sum_{j=1}^{n} \alpha_j^* k(x_i, x_j)\Big)^2 \\ &\quad + \lambda \sum_{i,j=1}^{n} \alpha_i^* \alpha_j^* \langle k(.,x_i), k(.,x_j)\rangle_{\mathcal{H}}. \end{aligned} \tag{9}$$

In vector form, this is a convex minimization problem

$$\arg\min_{\alpha} |S(w)^{\frac{1}{2}}(Y - K\alpha)|^2 + \lambda \alpha^\top K \alpha. \tag{10}$$

Therefore, α^* is obtained by taking the derivative of (10) with respect to α and setting it to zero.

$$\begin{aligned} & -K^\top S(w)^{\frac{1}{2}}\big(S(w)^{\frac{1}{2}}(Y - K\alpha^*)\big) + \lambda K \alpha^* = 0 \\ \rightarrow \quad & K^\top S(w) Y - K^\top S(w) K \alpha^* = \lambda K \alpha^* \\ \rightarrow \quad & \alpha^* = \big(K^\top S(w) K + \lambda K\big)^{-1} K^\top S(w) Y \\ \rightarrow \quad & \alpha^* = \Big(K^\top\big(S(w)K + \lambda I\big)\Big)^{-1} K^\top S(w) Y \\ \rightarrow \quad & \alpha^* = \big(S(w)K + \lambda I\big)^{-1} S(w) Y \end{aligned} \tag{11}$$

Appendix C
Derivation of $\frac{\partial H(W)}{\partial w_i}$

$$\begin{aligned} H(W) &= \frac{1}{\mathrm{Tr}\big(I - S(w)\big)} \Big(Y^\top \big(I - S(w)\big) Y \\ &\qquad - 2\Lambda^\top \big(I - S(w)\big) Y \\ &\qquad + \Lambda^\top \big(I - S(w)\big) \Lambda \Big) \end{aligned} \tag{12}$$

Take the derivative with respect to w_i for each term, and denote $s'(w_i) := s(w_i)(1 - s(w_i))$ we get

$$\begin{aligned} \frac{\partial}{\partial w_i} \frac{1}{\mathrm{Tr}\big(I - S(w)\big)} &= \frac{s'(w_i)}{\Big(\mathrm{Tr}\big(I - S(w)\big)\Big)^2} \\ \frac{\partial}{\partial w_i} Y^\top \big(I - S(w)\big) Y &= -y_i^2 s'(w_i) \\ \frac{\partial}{\partial w_i} - 2\Lambda^\top \big(I - S(w)\big) Y &= -2 \tfrac{\partial \Lambda^\top}{\partial w_i} \big(I - S(w)\big) Y \\ &\quad + 2\Lambda^\top \big(e_i s'(w_i) e_i^\top\big) Y \\ \frac{\partial}{\partial w_i} \Lambda^\top \big(I - S(w)\big) \Lambda &= \tfrac{\partial \Lambda^\top}{\partial w_i} \big(I - S(w)\big) \Lambda \\ &\quad - \Lambda^\top e_i s'(w_i) e_i^\top \Lambda \\ &\quad + \Lambda^\top \big(I - S(w)\big) \tfrac{\partial \Lambda}{\partial w_i}. \end{aligned} \tag{13}$$

To derive $\frac{\partial \Lambda}{\partial w_i}$, let $\Delta := \big(S(w)K + \lambda I\big)$.

$$\begin{aligned} \frac{\partial}{\partial w_i} \Lambda &= \frac{\partial}{\partial w_i} K \big(S(w)K + \lambda I\big)^{-1} S(w) Y \\ &= K \Big(\frac{\partial \Delta^{-1}}{\partial w_i} S(w) Y + \Delta^{-1} \frac{\partial S(w)}{\partial w_i} Y \Big) \\ &= K \Big(- \Delta^{-1} \big(e_i s'(w_i) e_i^\top K\big) \Delta^{-1} S(w) Y \\ &\quad + \Delta^{-1} e_i s'(w_i) e_i^\top Y \Big) \\ &= -K \Delta^{-1} e_i s'(w_i) e_i^\top \Lambda + K \Delta^{-1} e_i s'(w_i) e_i^\top Y \\ &= K \big(S(w)K + \lambda I\big)^{-1} e_i s'(w_i) \big(e_i^\top (Y - \Lambda)\big). \end{aligned} \tag{14}$$

Putting it all together, we have

$$\begin{aligned} \frac{\partial H(W)}{\partial w_i} &= \frac{s'(w_i)}{\Big(\mathrm{Tr}\big(I - S(w)\big)\Big)^2} \Big(\big(I - S(w)\big)^{\frac{1}{2}} (Y - \Lambda) \Big)^2 \\ &\quad + \frac{1}{\mathrm{Tr}\big(I - S(w)\big)} \Big(s'(w_i)(-y_i^2 + 2\Lambda_i y_i - \Lambda_i^2) \\ &\quad + \Big(2(\Lambda - Y)^\top \big(I - S(w)\big) \Big) \tfrac{\partial \Lambda}{\partial w_i} \Big). \end{aligned} \tag{15}$$

Appendix D
Vulnerability

Under certain circumstances, if the attacker creates more than one attack distribution, our method tends to have low recall and increased test MSE. Specifically, we demonstrate the special case where the attacker creates two attack distributions and show how both the poison ratio and the distance between distributions affect our method.

Consider the synthetic dataset introduced in Sec. IV-B. We perform two additive attacks, respectively with δ_{a_1} and δ_{a_2}, both of which tamper with the same amount of data. In Fig. 5, we adjust the distances between distributions by varying δ_{a_1} and δ_{a_2}. First, we let the poison ratio be 0.3, $\delta_{a_1} = 1$, $\delta_{a_2} = 2$, as shown in Fig. 5(a1). Fig. 5(a2) then shows that our method mistakes the poisons generated from δ_{a_1} for normal data but correctly classifies those generated from δ_{a_2}. Therefore, the precision and recall (p/r) are 100% and 50%. Fig. 5(a3) shows the resultant $f^*_{\bar{w}}$ and test MSE. We also show 1000 data respectively that have the smallest and largest residuals in $\mathcal{N}(\bar{w})$. This is because our bilevel formulation only assumes two distributions, one being the normal data and the other being the poisons. When there are two attack distributions, with one being sufficiently dense and close to the normal data, it is likely that our method will mistake them for normal data.

However, when the poison ratio is decreased to 0.2 while everything else is kept the same (Fig. 5(b1)), our method recovers the normal data with 100% precision and 99.9% recall as Fig. 5(b2) shows. The test MSE is also significantly reduced to 0.09 (Fig. 5(b3)). Our method thus shows robustness when an attacker can only tamper with less data, which results in sparser attack distributions.

Nevertheless, even with the attacker's limited access to tamper with data, the distance between attack distributions can still potentially be leveraged to mislead our method. In Fig. 5(c1), we keep the poison ratio at 0.2 but set $\delta_{a_1} = 0.5$ and $\delta_{a_2} = 2$. In Fig. 5(c2), it shows that our method mistakes again poisons from $\delta_{a_1} = 0.5$ for normal data, resulting in 50% recall. Yet, the decrease of poison ratio and δ_{a_1} lead to less effective poisons, the inclusion of which in $\mathcal{N}(\bar{w})$ to learn $f^*_{\bar{w}}$ increases the test MSE only to 0.39 (Fig. 5(c3)) as opposed to 3.19 (Fig. 5(a3)).

Fig. 5(d1) shows that the poison ratio is further decreased to 0.05. In this case, our method perfectly partitions the poisons from the normal data (Fig. 5(d2)), recovering the original test MSE 0.09 (Fig. 5(d3)).

References

[1] P. Kumar, Y. Lin, G. Bai, A. Paverd, J. S. Dong, and A. Martin, "Smart grid metering networks: A survey on security, privacy and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2886–2927, 2019.

[2] G. Dileep, "A survey on smart grid technologies and applications," *Renewable energy*, vol. 146, pp. 2589–2625, 2020.

[3] Y. Li and J. Yan, "Cybersecurity of smart inverters in the smart grid: A survey," *IEEE Transactions on Power Electronics*, 2022.

[4] C. Peng, H. Sun, M. Yang, and Y.-L. Wang, "A survey on security communication and control for smart grids under malicious cyber attacks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 8, pp. 1554–1569, 2019.

[5] B. Chen, J. Wang, X. Lu, C. Chen, and S. Zhao, "Networked microgrids for grid resilience, robustness, and efficiency: A review," *IEEE Transactions on Smart Grid*, vol. 12, no. 1, pp. 18–32, 2020.

[6] D. Wang, X. Wang, Y. Zhang, and L. Jin, "Detection of power grid disturbances and cyber-attacks based on machine learning," *Journal of information security and applications*, vol. 46, pp. 42–52, 2019.

[7] A. K. Jain, N. Sahani, and C.-C. Liu, "Detection of falsified commands on a der management system," *IEEE Transactions on Smart Grid*, vol. 13, no. 2, pp. 1322–1334, 2021.

[8] M. Goldblum, D. Tsipras, C. Xie, X. Chen, A. Schwarzschild, D. Song, A. Mądry, B. Li, and T. Goldstein, "Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 2, pp. 1563–1580, 2022.

[9] Z. Tian, L. Cui, J. Liang, and S. Yu, "A comprehensive survey on poisoning attacks and countermeasures in machine learning," *ACM Computing Surveys*, vol. 55, no. 8, pp. 1–35, 2022.

[10] I. Diakonikolas, G. Kamath, D. Kane, J. Li, J. Steinhardt, and A. Stewart, "Sever: A robust meta-algorithm for stochastic optimization," in *International Conference on Machine Learning*. PMLR, 2019, pp. 1596–1606.

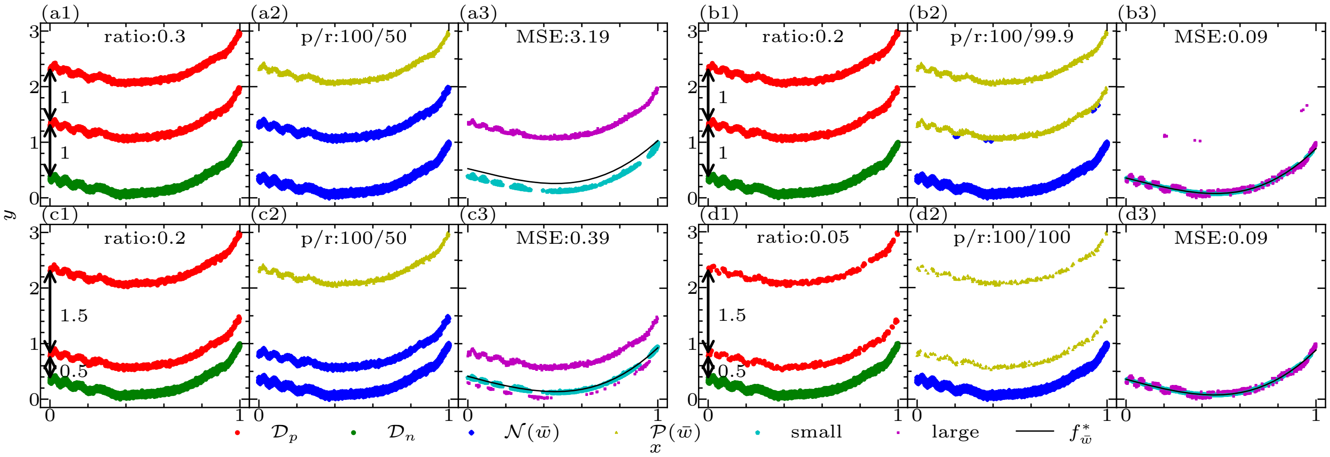

Fig. 5: **Illustration of vulnerability by a synthetic dataset.** (a) A poison ratio high enough with multiple distributions misleads our method. (b) Our method is robust when the poison ratio is lowered. (c) The distance between distributions is leveraged to mislead our method but with less influence. (d) Our method is robust when the poison ratio is further lowered.

[11] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," *arXiv preprint arXiv:1811.03728*, 2018.

[12] E. Chou, F. Tramer, and G. Pellegrino, "Sentinet: Detecting localized universal attacks against deep learning systems," in *2020 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2020, pp. 48–54.

[13] J. Jia, X. Cao, and N. Z. Gong, "Intrinsic certified robustness of bagging against data poisoning attacks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 9, 2021, pp. 7961–7969.

[14] Y. Ma, X. Zhu, and J. Hsu, "Data poisoning against differentially-private learners: Attacks and defenses," *arXiv preprint arXiv:1903.09860*, 2019.

[15] Y. Zeng, M. Pan, H. Jahagirdar, M. Jin, L. Lyu, and R. Jia, "How to sift out a clean data subset in the presence of data poisoning?" *arXiv preprint arXiv:2210.06516*, 2022.

[16] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *2018 IEEE symposium on security and privacy (SP)*. IEEE, 2018, pp. 19–35.

[17] N. Müller, D. Kowatsch, and K. Böttinger, "Data poisoning attacks on regression learning and corresponding defenses," in *2020 IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2020, pp. 80–89.

[18] P. J. Huber, "Robust estimation of a location parameter: Annals mathematics statistics, 35," *Ji, S., Xue, Y. and Carin, L.(2008),'Bayesian compressive sensing', IEEE Transactions on signal processing*, vol. 56, no. 6, pp. 2346–2356, 1964.

[19] C. Yu and W. Yao, "Robust linear regression: A review and comparison," *Communications in Statistics-Simulation and Computation*, vol. 46, no. 8, pp. 6261–6282, 2017.

[20] S. Karmalkar, A. Klivans, and P. Kothari, "List-decodable linear regression," *Advances in neural information processing systems*, vol. 32, 2019.

[21] H. Zhang, B. Liu, and H. Wu, "Smart grid cyber-physical attack and defense: A review," *IEEE Access*, vol. 9, pp. 29 641–29 659, 2021.

[22] G. Prasad, Y. Huo, L. Lampe, and V. C. Leung, "Machine learning based physical-layer intrusion detection and location for the smart grid," in *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2019, pp. 1–6.

[23] Y. Arjoune, F. Salahdine, M. S. Islam, E. Ghribi, and N. Kaabouch, "A novel jamming attacks detection approach based on machine learning for wireless communication," in *2020 International Conference on Information Networking (ICOIN)*. IEEE, 2020, pp. 459–464.

[24] J. Yeckle and B. Tang, "Detection of electricity theft in customer consumption using outlier detection algorithms," in *2018 1st international conference on data intelligence and security (ICDIS)*. IEEE, 2018, pp. 135–140.

[25] A. Jindal, A. Dua, K. Kaur, M. Singh, N. Kumar, and S. Mishra, "Decision tree and svm-based data analytics for theft detection in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 3, pp. 1005–1016, 2016.

[26] M. Ismail, M. F. Shaaban, M. Naidu, and E. Serpedin, "Deep learning detection of electricity theft cyber-attacks in renewable distributed generation," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3428–3437, 2020.

[27] A. Takiddin, M. Ismail, R. Atat, K. R. Davis, and E. Serpedin, "Robust graph autoencoder-based detection of false data injection attacks against data poisoning in smart grids," *IEEE Transactions on Artificial Intelligence*, 2023.

[28] A. Takiddin, M. Ismail, U. Zafar, and E. Serpedin, "Robust electricity theft detection against data poisoning attacks in smart grids," *IEEE Transactions on Smart Grid*, vol. 12, no. 3, pp. 2675–2684, 2020.

[29] M. Billah, A. Anwar, Z. Rahman, and S. M. Galib, "Bi-level poisoning attack model and countermeasure for appliance consumption data of smart homes," *Energies*, vol. 14, no. 13, p. 3887, 2021.

[30] S. Bhattacharjee, M. J. Islam, and S. Abedzadeh, "Robust anomaly based attack detection in smart grids under data poisoning attacks," in *Proceedings of the 8th ACM on Cyber-Physical System Security Workshop*, 2022, pp. 3–14.

[31] C. A. Micchelli, Y. Xu, and H. Zhang, "Universal kernels." *Journal of Machine Learning Research*, vol. 7, no. 12, 2006.

[32] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[33] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *Computational Learning Theory: 14th Annual Conference on Computational Learning Theory, COLT 2001 and 5th European Conference on Computational Learning Theory, EuroCOLT 2001 Amsterdam, The Netherlands, July 16–19, 2001 Proceedings 14*. Springer, 2001, pp. 416–426.

[34] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," *Advances in neural information processing systems*, vol. 20, 2007.

[35] W. Rudin, *Fourier analysis on groups*. Courier Dover Publications, 2017.

[36] M. A. Woodbury, *Inverting modified matrices*. Department of Statistics, Princeton University, 1950.

[37] V. Arzamasov, "Electrical Grid Stability Simulated Data ," UCI Machine Learning Repository, 2018, DOI: https://doi.org/10.24432/C5PG66.

[38] L. Candanedo, "Appliances energy prediction," UCI Machine Learning Repository, 2017, DOI: https://doi.org/10.24432/C5VC8G.

[39] P. Tfekci and H. Kaya, "Combined Cycle Power Plant," UCI Machine Learning Repository, 2014, DOI: https://doi.org/10.24432/C5002N.

[40] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[41] W. H. Greene, *Econometric analysis*. Pearson Education India, 2003.

[42] X. Qi, T. Xie, Y. Li, S. Mahloujifar, and P. Mittal, "Revisiting the assumption of latent separability for backdoor defenses," in *The eleventh international conference on learning representations*, 2022.