AMPLIFYING PRIVACY ATTACKS ON TABULAR FOUNDATION MODELS VIA A STOLEN SURROGATE

Anonymous authors
Paper under double-blind review

ABSTRACT

Tabular foundation models (TFMs) achieve state-of-the-art performance on structured data. The fact that these models are usually adapted for their sensitive downstream tasks by in-context learning raises the question of privacy risks. We show that even a query-only adversary can extract a significant amount of private information from the context examples by first stealing a surrogate of the preconditioned model and then running privacy attacks on the copy. Without a single sample from the in-context data distribution, without knowledge of the context size, and without any write access to the context, the adversary sends random queries, adapts a public TFM checkpoint to the answers, and obtains a high-fidelity surrogate. The adversary can then condition on arbitrary contexts and query the surrogate without any restrictions. We show that the surrogate can significantly amplify the risks of standard privacy attacks, including membership inference, attribute inference, and property inference. We analyze our method across various settings, including different datasets, TFM backbones, context sizes, and output formats. We find that the combination of stealing and privacy attacks generalizes over these settings, and that its success grows with task difficulty. Restricting the output to the predicted label is still vulnerable to theft, but prevents privacy attacks on a classification task, whereas private information from regression still leaks. Our anonymous codebase is available at `https://anonymous.4open.science/r/tfm-priv-attack-0BFC`.

1 INTRODUCTION

Tabular foundation models (TFMs) have become state-of-the-art for prediction on structured data. They are pretrained to learn general patterns in the data and can then be adapted to particular downstream tasks through in-context learning (Müller et al., 2022; Hollmann et al., 2023; Grinsztajn et al., 2025; Qu et al., 2025; Ma et al., 2025). This approach has been shown to outperform gradient-boosted trees and strongly tuned baselines fitted to each dataset individually (Erickson et al., 2025).

Following developments from other domains, such as vision (Clarifai) and language (Cohere; OpenAI), TFMs are increasingly deployed behind paid query APIs. In addition to accessing the base models, downstream providers also have the possibility to provide query access to adapted models, *i.e.,* models that carry the provider's labeled downstream data in the context (Yeh et al., 2025; PriorLabs, 2025). Users can then interact with these adapted models through black-box query access.

However, since downstream data is often sensitive, this practice raises concerns about privacy risks. Initial prior work has explored the success of membership inference attacks (Shokri et al., 2017) against the in-context downstream data (Carvalho & Cordy, 2026). These attacks aim to determine whether a given record was used for training, or in the case of TFMs, included in the context. Under the assumption that an adversary has access to the model internals and real member or non-member data, or at least data from the exact downstream distribution, these attacks have been shown to be successful. However, under realistic assumptions, where adversaries only have black-box query access and no auxiliary data, the attack success converges towards random guessing (Wahdany et al., 2026). This might suggest that the in-context data is still private enough, *in practice*.

In this work, we show that this assumption is too optimistic and that even with black-box only access and without any in-distribution data, an adversary can leak significant amounts of private information from the in-context data. Therefore, we explore an attack vector that has not been explored against

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

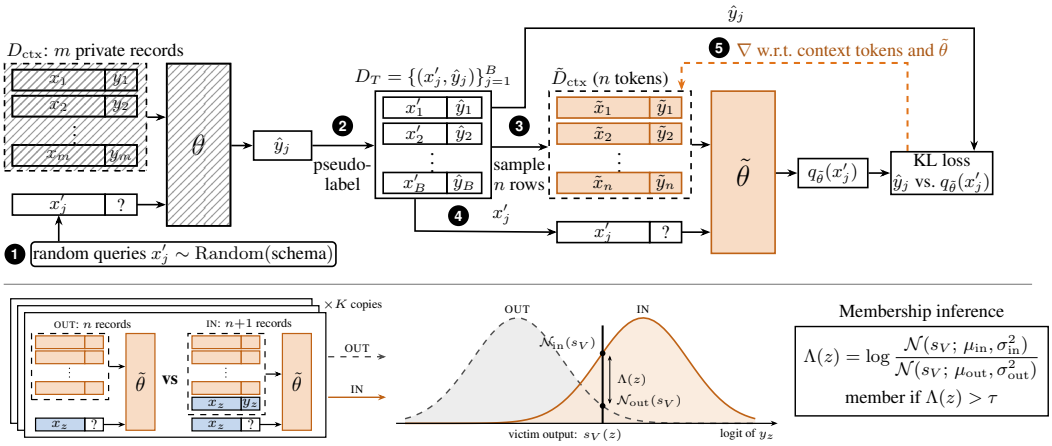


Figure 1: **Stealing the victim TFM, then inferring membership through the surrogate.** *Top:* the adversary queries the victim with random rows drawn from the public schema and fits the context and weights of its base model (orange) to reproduce the victim's predictions, without ever observing D_{ctx} or θ (hatched). *Bottom:* an example of how the stolen copy can be utilized, here for membership inference. Each of K surrogates is read at the target record z twice, with its own n records (OUT) and with z appended (IN), and the victim's answer at x_z is calibrated against the resulting distributions.

TFMs, namely one where the adversary first extracts the target model, obtains a stolen local surrogate, and then runs privacy attacks locally (Tramèr et al., 2016). To the best of our knowledge, we are the first to explore stealing and its effect on downstream privacy for in-context TFMs. We first show that by querying with randomly generated rows alone, an adversary can obtain a stolen surrogate that reaches a fidelity of 0.949 to the deployed target model on a held-out evaluation set. Next, we adapt the most common privacy attacks, including membership inference (Carlini et al., 2022; Ye et al., 2022; Zarifzadeh et al., 2024), attribute inference (Fredrikson et al., 2014; Yeom et al., 2018), and property inference (Ateniese et al., 2015; Zhang et al., 2021; Suri & Evans, 2022) to the stolen surrogates and measure their success against conducting the attack against the black-box deployed TFM directly (non-stealing baseline) and a white-box version of the adapted model (oracle baseline).

We perform our evaluation on three state-of-the-art TFMs, namely TabPFN-3 (Grinsztajn et al., 2026), RealTabPFN-2.5 (Grinsztajn et al., 2025), TabICLv2 (Qu et al., 2026) using TabArena (Erickson et al., 2025). We show that for membership inference, our attack achieves a true positive rate (TPR) of 35.9% at a 5% false positive rate (FPR), against 21.7% for the non-stealing baseline, and reaches two-thirds of the 54.0% of the oracle baseline (Table 1). For attribute inference that aims at recovering individual attributes of in-context rows, our attack recovers 66.9% of the records when one attribute is unknown and 34.5% when five are, roughly doubling the 37.2% and 18.0% of the non-stealing baseline. The results for the five unknown attributes even closely match the oracle baseline, which additionally requires write access to the target model's context. For property inference, we aim to estimate the label composition of the in-context data. We achieve a mean squared error of 0.055 against 0.071 for the non-stealing baseline over subgroups of the context that share the same values on three given attributes.

Table 1: **Membership inference.** averaged over the datasets, with 32 copies.

	Baseline	Ours	Oracle
TPR@5%FPR	0.217	**0.359**	0.540
AUC	0.705	**0.750**	0.875

We show that the approach of first stealing an adapted TFM also increases the success of subsequent privacy attacks if the adversary does not have access to the same backbone as the target model, and does not use the same context-size. The privacy leakage increase grows with task difficulty. Finally, we analyze the impact of restricting the outputs from the deployed TFM on the attack success. We find that for classification, returning only the top-1 label still permits stealing while mitigating the success of the privacy attacks which for regression, the final estimate still leaks.

In summary, we make the following contributions:

- We explore a new attack vector against deployed TFMs that first steals the adapted target model and then performs privacy attacks against the local surrogate.
- We show that TFMs can be easily stolen, even when an adversary only has black-box access to the model and no in-distribution data and queries with random rows instead.
- We find that this attack setup is highly effective to leak private information about the deployed in-context data, more than doubling the success of attacks that have much stronger assumptions, ran against the API.
- We analyze the success factors of the attack, including knowledge of the backbone and in-context size, and show that even with limited knowledge and with mitigations in place, an adversary can still successfully learn private information.

2 Background and Related Work

Tabular Foundation Models. TFMs use in-context learning to solve tabular prediction tasks: Given a labeled table D_{ctx} placed in their context and an unlabeled query x_q, they produce, in a single forward pass and without any gradient updates, a distribution $q_\theta(y \mid x_q, D_{\text{ctx}})$ that approximates the posterior predictive over the label induced by their pretraining prior. They are pretrained on tables sampled from a synthetic prior following the prior-data fitted network paradigm (Müller et al., 2022). Hollmann et al. (2023) demonstrated that this approach allows solving small tabular classification problems in seconds, and successive generations have extended it to missing values, regression, and heterogeneous column types (Hollmann et al., 2025), decoupled the in-context stage from the number of features (Qu et al., 2025), and scaled contexts to one million rows (Grinsztajn et al., 2026; Qu et al., 2026). Since state-of-the-art TFMs are exclusively trained on synthetic data (Grinsztajn et al., 2026; Qu et al., 2026; Kong & Das, 2026; Tao et al., 2026; Eo et al., 2026), no real record entered their parameters. Variants that continue pretraining on real corpora exist (Ma et al., 2025; Grinsztajn et al., 2025), but for the synthetically pretrained models the only privacy-relevant data is the table placed in the context, which is what we attack. Every attack below therefore targets D_{ctx} and not θ.

Model Stealing. Model stealing attacks try to extract the functionality of a machine learning model that is only provided to the public via limited access, returning only labels, confidence scores, or generated outputs for a given input. The main mechanism behind most of these attacks is to use the target model to label data points provided by the attacker, and to train the surrogate model on the result (Tramèr et al., 2016; Orekondy et al., 2019; Papernot et al., 2017). Success is measured by *fidelity*, the agreement between the surrogate's and the target's outputs, most often measured as the rate at which their top-1 labels agree (Tramèr et al., 2016; Jagielski et al., 2020). While most of these attacks have been developed and evaluated on classifier models, the concept has in recent times been extended to foundation models, such as Large Language Models (LLMs) (Krishna et al., 2020; Carlini et al., 2024). As of the writing of this paper, no dedicated model stealing attacks for TFMs have been proposed. However, the mechanisms of the original, classifier-based attacks are transferable to these models since a TFM with a given context can be used to assign pseudo-labels to unlabeled rows provided by the attacker.

Membership Inference. Membership inference attacks (MIA) aim to determine whether a single data point was part of the training data of an ML model (Shokri et al., 2017). These attacks are based on certain signals a model produces when operating on an input, such as confidences, losses, or output labels (Yeom et al., 2018; Song & Mittal, 2021; Choquette-Choo et al., 2021). Signals are compared between known member and non-member data, and the attacker can accordingly classify the input in question. State-of-the-art attacks use shadow models trained on strictly controlled sets of in-distribution training data that differ from model to model, which allows them to approximate the member and non-member distributions (Carlini et al., 2022; Ye et al., 2022; Zarifzadeh et al., 2024). This requires data from the victim's distribution. Proposed methods aimed at TFMs make additional assumptions about attacker capabilities, such as access to a dataset of the same distribution as the victim model's context, or limited abilities to alter the target model (Carvalho & Cordy, 2026; Wahdany et al., 2026; Jälkö et al., 2026). Our methods do not assume such capabilities, working with pure query access to the victim model, and no information on the context data.

Attribute Inference. Attribute inference attacks (AIA) aim to recover a specific attribute value in a data record, with the other attributes already known by the attacker (Yeom et al., 2018; Fredrikson et al., 2014). These attacks rely on querying the model with versions of the record with the attribute

in question filled in with all candidate values, and comparing output signals such as confidence or loss (Fredrikson et al., 2014). Attribute inference may also be used to recover multiple attributes iteratively by conducting the attack on a given attribute and adding the recovered value to the record in future attacks (Protivash et al., 2024). While these attacks assume a tabular training dataset, they predate TFMs. To the best of our knowledge, no dedicated attribute inference attacks have been proposed for TFMs.

Property Inference. Property inference attacks (PIA) are concerned with deriving broader properties of the training dataset of the target model, as opposed to specific data samples (Ateniese et al., 2015; Ganju et al., 2018; Suri & Evans, 2022). The most commonly attacked properties are the percentages of data samples with specific feature values within the dataset. Since, for TFMs, the context does not have any influence on the model's weights, shadow models must be differentiated by their query behavior. These attacks require in-distribution data to assemble the shadow training sets, which a query-only adversary cannot obtain.

3 The Attack

Threat Model. We write θ for the pretrained backbone the provider serves and D_{ctx} for the private table of m records they place in the context. On a query x_q, the target model returns $q_\theta(y \mid x_q, D_{\mathrm{ctx}})$ in a single forward pass, with no gradient step and no change to θ. Like any benign user, the adversary knows the required schema, including column names, types, categorical levels, and the declared range of numeric columns. However, they do not know the data distribution, which TFM model is being used, or the context size m, and cannot modify D_{ctx}. We assume that the adversary has their own TFM backbone $\tilde{\theta}$, which may or may not be the same TFM as θ. The attack proceeds in two steps. (1) The adversary first steals the functionality of the adapted black-box TFM to obtain a local surrogate. (2) The adversary runs privacy attacks against this surrogate to leak information on the in-context data.

3.1 Functionality Stealing

The target of stealing is not the TFM backbone alone but the backbone preconditioned on its private context. The stealing stage consists of five steps, depicted in Figure 1.

(1) The adversary draws B random queries x'_j that follow the public schema.
(2) It sends each one to the target model and obtains the full prediction $\hat{y}_j$ it returns. The pairs of query and answer form the transfer set D_T used to obtain the local surrogate.
(3) It gives $\tilde{\theta}$ a starting context of n rows randomly selected from the labeled D_T.
(4) It queries this surrogate with the x'_j in D_T.
(5) It optimizes the surrogate's context and weights to minimize the KL divergence between the surrogate's answer and $\hat{y}_j$ on each row.

To optimize the context in step (5), we first map the initial context into the **embedding space** and optimize the embedding via gradient descent. The resulting embedding may not correspond to an actual context table, but it can be used in the stolen TFM (which operates with queries in the data space, not in the embedding space). To get a real optimal context table, we can map the embedding back into the **data space**: Each numeric cell is adjusted to fit into its declared range, and each categorical cell is rounded to a discrete value. This results in a table of records that could be served directly as the context of a TFM.

Repeating steps (3)–(5) from K different starting contexts gives K surrogate copies (or *shadow models*), without additional target model queries, for use in the subsequent privacy attacks.

3.2 Privacy Attacks

After stealing the preconditioned target model's functionality into one or multiple local surrogate models, the adversary can conduct privacy attacks to leak information on the private in-context data.

Membership Inference. When accessing only the black-box target model, an adversary without knowledge of member and non-member data cannot obtain shadow models that support state-of-the-

art strong membership inference attacks (Carlini et al., 2022; Ye et al., 2022; Zarifzadeh et al., 2024). In our setup, such shadow models can be obtained, as described above. Formally, let $s_V(x)$ denote the target model's prediction at a query x for the target record's label y_z, translated to a logit by the adversary, and $s_S(x)$ be the surrogate prediction. The surrogates provide us with the OUT statistics $\mu_{\text{out}}, \sigma_{\text{out}}$, the mean and standard deviation of $s_S(z)$ over the copies: what a model of this function answers at z when z is not in its context. When we append z to the surrogate's own context, they give the IN statistics $\mu_{\text{in}}, \sigma_{\text{in}}$. The two Gaussians these statistics define are the OUT and IN curves in the middle of the row. We score

$$\Lambda(z) \;=\; \log \mathcal{N}\big(s_V(z);\, \mu_{\text{in}}, \sigma_{\text{in}}^2\big) - \log \mathcal{N}\big(s_V(z);\, \mu_{\text{out}}, \sigma_{\text{out}}^2\big)\,, \tag{1}$$

which asks which of the two hypotheses better explains what the target model returned. In Figure 1, the target model's output $s_V(z)$ is the vertical line, and $\Lambda(z)$ compares the heights of the two curves where the line crosses them. We assume z is a member if $\Lambda(z)$ exceeds a threshold τ. §B.1 describes how each copy can be refined on the neighbor records of z for better performance.

Attribute Inference. In attribute inference, the attacker has partial knowledge of the data (Fredrikson et al., 2014), i.e., they know some *anchor* attributes A and the label y_z. They want to learn the set of unknown attributes U. Plausible candidates for U can be obtained from the public schema of the deployed adapted TFM. For categorical attributes, these are the possible categories, for numeric attributes, they can be sampled from the possible ranges. Among all of these candidates, the attribute inference attack then ranks the most likely one. Therefore, it relies on the observation that a TFM is more confident on data records that were in its context than on records that were not. We again rely on the shadow models.

However, attribute inference requires a finer signal than membership inference. Membership inference compares the target record with other records, which generally lie far apart in feature space. Attribute inference instead compares candidates that share the anchor and differ only in the values of U, i.e., records that lie close to each other, and on which the target model's answers differ only slightly. Rather than reading only the confidence in the label y_z, as membership inference does (Eq. (1)), we therefore use the full predictive distribution. Concretely, we model as $q_V(c \mid x)$ the probability of class c produced by the target model given the features x and $q_S(c \mid x)$ the mean probability produced by the surrogates. Then, $\Delta_c(x) = \operatorname{logit} q_V(c \mid x) - \operatorname{logit} q_S(c \mid x)$ is the difference in confidence between the target model and the surrogates. We score the candidates $x_{A,U}$ with fixed anchor attributes A and free attributes U for attribute inference by the full-distribution score

$$\text{FD}(x_{A,U}) \;=\; \Delta_{y_z}(x_{A,U}) \;-\; \frac{1}{|\mathcal{Y}|-1} \sum_{c \neq y_z} \Delta_c(x_{A,U}), \tag{2}$$

where $\mathcal{Y}$ is the set of labels. FD increases when the target model is more confident than the surrogates in y_z and when it is less confident in the other labels. The attack predicts the values of U of $\arg\max_{x_{A,U}} \text{FD}(x_{A,U})$.

The computation of FD costs one query to the target model per candidate, and q_S is read on the adversary's own surrogates at no cost to the target model.

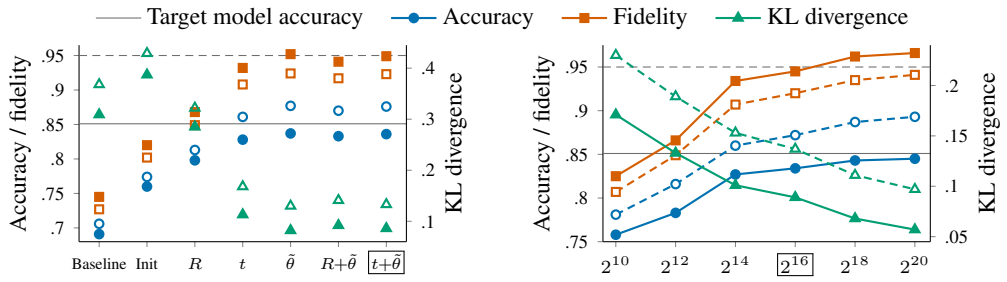
Property Inference. Property inference aims to determine the composition of the entire in-context data, rather than properties of individual rows. Given a region $R = \{x : x_{c_1} = v_1, \ldots, x_{c_\ell} = v_\ell\}$, defined by ℓ equality conditions which specify values for categorical columns, and a class c, the adversary estimates

$$\alpha(R, c) = P(y = c \mid x \in R), \tag{3}$$

which is the percentage of the private records in region R that have label c. With $\ell = 0$, R is the whole in-context data, and thus, α is the label distribution of the entire context.

The adversary first randomly draws N_R probe records from R using the public schema. They then query the target model once on each probe and take the target model's top prediction as the probe's pseudo-label $\hat{y}(x)$. These are the only target model queries the attack requires. Each probe x is then scored at its own $\hat{y}(x)$ by the percentage of the K surrogate copies that have a lower confidence for this label than the target model. We establish

$$\text{RANK}(x) \;=\; \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}[\, s_V(x) > s_{S_k}(x) \,]\,, \tag{4}$$



(a) Optimization target (b) Query budget B

Figure 2: **STEALING: optimization target and query budget.** Each point is the mean over the datasets. Across both panels, solid lines and filled markers indicate results on held-out data; dashed lines and open markers indicate results on the context members. The grey horizontal lines are the target model's accuracy, which a perfect surrogate would match. **(a)** Optimization is restricted to certain components of the surrogate. Targets: context (record-space R, or token-space t), weights $\tilde{\theta}$, or both ($R{+}\tilde{\theta}$, $t{+}\tilde{\theta}$). *Init* is the initial surrogate without optimization. Query budget is $B = 2^{16}$. **(b)** Query budget B determines the size of the transfer set D_T. The optimization target is $t{+}\tilde{\theta}$. Boxed x-axis values mark the defaults used in every other experiment.

where $s_V(x)$ and $s_{S_k}(x)$ are the logits the target model and the k-th surrogate copy assign to $\hat{y}(x)$, respectively. RANK asks how the target model's answer compares to the copies' answers, without fitting a distribution over them. The top 10% of probes according to RANK are kept, with ties broken at random, and the fraction of remaining probes with pseudo-label c is reported as the estimate of $\alpha(R, \cdot)$. This pruning step is crucial, since randomly generated records often lie far from every record in the context. For such probes, the victim's answer is unconstrained by the context, and thus, their assigned pseudo-labels carries no information about it.

4 EXPERIMENTS

Models. For our main experiments, we use TabICLv2 for θ and $\tilde{\theta}$, with matching context sizes, namely 256. We also evaluate the effect of choosing different TFM architectures, including TabPFN-3 and TabICLv2, and different context sizes in §4.2. We provide further details on the setup in §A.

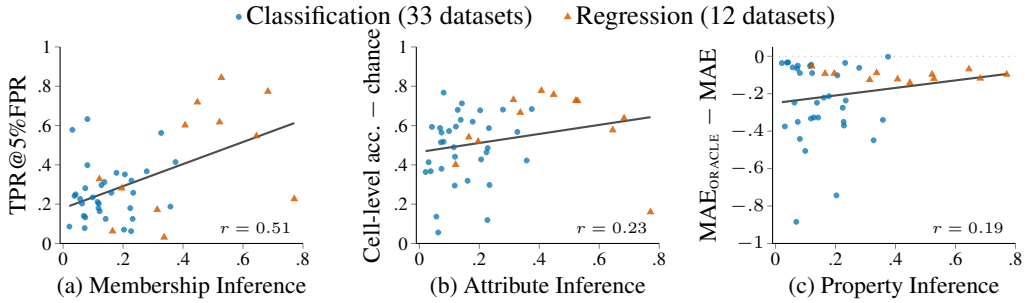
Data. We evaluate on TabArena (Erickson et al., 2025), a popular benchmark of 51 datasets, 38 classification and 13 regression. From these, we chose the subset of datasets with fewer than 100 features, which reduces the total number to 45, with 33 classification and 12 regression tasks.

API Deployment and Queries All attacks operate on the vector of class probabilities. Classifiers return these directly; however, regressors only return a predictive distribution over the target. For regression, the adversary therefore divides the target range declared in the public schema into 10 fixed bins and the probability mass corresponds to the predicted distribution assigned to each bin. A single forward pass returns the full predictive distribution, so each queried row still costs exactly one query, regardless of the number of bins. We describe this further in §B.4.

Baselines. We compare the success of our privacy attacks against two baselines. The first one gives an intuition on the lower bound on the success by running the same attack type against the adapted target TFM behind the API directly. We refer to it as the *non-stealing baseline*. The other one gives an intuition on the upper bound on success by running a white-box version of the attack directly against the adapted target TFM. We call this our *oracle baseline*. To have a strong upper bound, we equip the oracle baselines also with other capabilities that our attacks do not have: For membership and property inference, we let the oracle use *real records* from the target model's data distribution. For attribute inference, we let the oracle attack insert manipulated records into the target model's context.

4.1 ATTACK EVALUATION

Functionality stealing (STEALING). We first evaluate the success of our stealing attack. Therefore, we measure (1) *fidelity*, i.e., the rate at which the surrogate's top-1 label matches the target model's, (2) *accuracy* of the surrogate, and (3) the *Kullback-Leibler divergence* (KL-divergence) from the



(a) Membership Inference (b) Attribute Inference (c) Property Inference

Figure 3: **Leakage against task difficulty.** One point per TabArena dataset. The x-axis is the victim's held-out error, $1-$accuracy, representing the difficulty of a dataset. For regression, the target and output are discretized into 10 cells for easier comparison to classification. In (b), the random baseline is derived by randomization over the unknown attribute's values. In (c), the y-axis is the error of the ORACLE method minus the error of ours (higher is better), with 0 marking parity with the ORACLE. Each solid line is one least-squares fit over all datasets. r is the Pearson correlation over all points.

target's posterior to the surrogate's. We report these on both a held-out evaluation set and the private context data. For the baseline, we use a nearest-neighbor lookup over D_T, which answers each query with the target model's ground truth label from the closest record.

We present our results in Figure 2. Figure 2(a) varies which parts of the surrogate are optimized. Every choice improves on the unoptimized surrogate (denoted *Init* with fidelity 0.820 on held-out data) and on the lookup (denoted *baseline* with fidelity 0.75 on held-out data). With the weights fitted alone ($\tilde{\theta}$), fidelity reaches 0.952. Adding a context changes fidelity by at most 0.011. Among context-only fits, the embedding-space context (t, 0.932) improves the data-space (R, 0.868), because the data-space context is bound to valid records (§3.1). Based on the results, we use $t{+}\tilde{\theta}$ (0.949) by default. Figure 2(b) varies the query budget B. Fidelity on held-out rows rises quickly at small budgets, from 0.825 at $B = 2^{10}$ to 0.934 at 2^{14}, and then flattens, gaining only 0.032 more up to 2^{20}. The in-context members follow the same trend but with slightly lower values. At the default $B = 2^{16}$, fidelity is 0.945 and the surrogate's accuracy (0.834) is within 0.017 of the target model's (0.851). Across both experiments, the values on the private in-context data are slightly lower than on the general held-out data. We hypothesize that the target model represents the records in its context better than other in-distribution data, which random queries rarely reach. Despite this gap, the next sections show that stealing a surrogate first substantially improves the privacy attacks.

Membership inference. We evaluate on a balanced pool of 128 context members and 128 non-members, reporting TPR@5%FPR and AUC. The non-stealing baseline uses no surrogate. Instead, it scores z by how much the target model's confidence in y_z at z exceeds its average confidence at the neighbors of z that differ in one attribute. The oracle baseline builds each reference by providing in-distribution data from the same table directly to the target model. To close the gap between the surrogate and target near z, the attacker goes through a refinement stage that fine-tunes each surrogate with z's neighbor records. The adversary queries the target model on $B/2$ records close to z, namely all records that differ from z in one or two attributes, then fine-tunes each copy for one epoch on these answers. All copies share these answers, so the refinement costs $B/2$ target model queries per target record, independently of K. These records are generated from z and the public schema alone, so the refinement stays within the threat model. §B.1 compares refined and unrefined copies.

We use $K = 32$ surrogates throughout the attacks; §B.2 shows how the attack changes with K. Figure 3(a) also reports each dataset's membership leakage against the difficulty of its prediction task, measured as the target model's held-out error. The slope is positive, which indicates that harder tasks leak more. The other attacks depicted in Figure 3(b) and (c) show a weaker but similar trend.

Attribute inference. Since rows sharing an anchor are indistinguishable to an adversary, we count any recovered record as a correct match if it corresponds to a record in the target model's private in-context data. We report this as the record-level accuracy. Additionally, we also report a per-cell accuracy which is always scored against the true target record. The NON-STEALING baseline is the same target model-only output score to identify the candidate attribute(s) with highest confidence. The

Table 2: **Attribute inference and property inference performance. (a)** $|U|$ is the number of unknown attributes to recover. COV is the percentage of target records contained in the candidate set, which caps the record-level accuracy. **(b)** ℓ is the number of conditions that determine the target region. #TAB is the number of tables with a region of the given ℓ.

$\lvert U\rvert$	COV	NON-STEALING	AIA	ORACLE
1	.982	.372 / .372	.669 / .669	.793 / .793
2	.948	.307 / .146	.624 / .458	.731 / .611
3	.395	.198 / .037	.487 / .191	.510 / .289
4	.185	.186 / .014	.399 / .091	.418 / .151
5	.093	.180 / .012	.345 / .044	.346 / .084

(a) Attribute Inference (per-cell / per-record accuracy)

ℓ	#TAB	NON-STEALING	PIA	ORACLE
0	45	.234 / .101	.220 / .092	.016 / .001
1	29	.207 / .077	.182 / .061	.077 / .012
2	26	.210 / .074	.188 / .061	.070 / .011
3	24	.209 / .071	.177 / .055	.097 / .021

(b) Property Inference (MAE / MSE)



(a) STEALING (fidelity) (b) MIA (TPR@5%FPR) (c) AIA (cell-wise acc.) (d) PIA (MAE)

(e) STEALING (fidelity) (f) MIA (TPR@5%FPR) (g) AIA (cell-wise acc.) (h) PIA (MAE)

Figure 4: **Backbone and context-size mismatch impact. (a)–(d)** Row is the victim backbone θ, and column is the surrogate checkpoint $\tilde{\theta}$ the attacker holds; the diagonal is the matched case. **(e)–(h)** Row is the victim context size m, and column is the surrogate prompt size n chosen by the attacker. Darker color indicates stronger leakage throughout all matrices.

ORACLE baseline is adapted from a state-of-the-art attack called Truth-Serum (Tramèr et al., 2022). The attack appends mislabeled copies of each candidate to D_{ctx} (a capability that our stealing-based attack does not have) and analyzes the target model's confidence (details in §B.3). The intuition is that inserting copies of the target point with a wrong label pushes the model to predict that wrong label whenever the real point is absent. Therefore, any noticeable confidence in the correct label can only come from the real point being in the in-context data.

Table 2a varies the number of unknown attributes $|U|$. Per-cell accuracy of FD (see Eq. (2)) is 0.669 against 0.372 for the NON-STEALING baseline with one unknown attribute, and 0.345 against 0.180 with five. FD thus roughly doubles the baseline's accuracy at every $|U|$. Its gap to the ORACLE closes as $|U|$ grows, from 0.124 at $|U| = 1$ to 0.001 at $|U| = 5$. Record-level accuracy falls faster, mostly because our candidate set is capped at 256. Hence, from $|U| = 3$ on, the true record is in the set for fewer than 40% of targets (COV), which limits every attack, including the ORACLE.

Property inference. We selected the most frequent combination of features in D_{ctx} as the challenge regions, with $\ell \in \{0, 1, 2, 3\}$ conditions over the selected columns, where $\ell = 0$ is the entire context.

In Table 2b, we report the mean absolute error (MAE) and mean squared error (MSE) of $\hat{\alpha}$ against α, separately for each number of conditions ℓ. The NON-STEALING baseline skips the pruning step of §3.2: it keeps all N_R probes and reports the composition of their pseudo-labels. It thus makes the same target model queries as our attack, but uses no surrogate copies. The ORACLE reports the label distribution of the full dataset, i.e., of all rows of the table and not only those in the context, and uses it as the estimate for every region.

Our stealing-based property inference improves on the baseline for every ℓ, in both MAE and MSE. The MAE falls from 0.234 to 0.220 for the whole context ($\ell = 0$) and from 0.209 to 0.177 for regions with three conditions, a relative reduction of 6% and 15%. The gain is modest, but it grows as the regions become narrower. The ORACLE remains more accurate, but its advantage in MAE shrinks from 0.204 at $\ell = 0$ to 0.080 at $\ell = 3$.

Table 3: **Impact of output granularity on each attack.** STEALING reports fidelity to the victim in (a) and the MAE to the victim's scalar answer in (b). MIA reports TPR@5%FPR and AUC, AIA reports per-cell accuracy with two undisclosed attributes, and PIA reports the MAE of the context's label shares within regions defined by two conditions. **Bold**: attack performs no better than baseline.

	STEALING	MIA		AIA	PIA
Output	Fid	$\text{TPR}_{5\%}$	AUC	Acc	MAE
Full	.874	.484	.866	.620	.188
Top-1+conf.	.870	.391	.864	.565	.132
Top-1	.855	**.055**	**.497**	**.160**	**.357**

(a) Classification (`students_dropout`)

	STEALING	MIA		AIA	PIA
Output	MAE	$\text{TPR}_{5\%}$	AUC	Acc	MAE
Full	.152	.719	.880	.685	.044
Scalar	.151	.668	.941	.285	.048

(b) Regression (`houses`)

4.2 ABLATIONS

Knowledge of the backbone and context-size. By default, the adversary uses the same backbone as the target model and sets its prompt size to the context size. Figure 4 ablates both choices. With a different base model (a–d), each attack's primary metric changes by at most 0.018 on average. The type of the target model matters far more, with MIA's TPR@5%FPR at 0.133–0.188 against RealTabPFN-2.5 and 0.883–0.945 against TabPFN-3.

We observed similar results with a wrong prompt size n (e–h). It changes the stealing fidelity by at most 0.007 in any cell and the attribute accuracy by less than 0.01 on average, and it does not lower MIA's TPR@5%FPR on average. PIA shows no consistent trend in m or n, with its MAE between 0.130 and 0.214. Leakage decreases as the target model's context grows, but at $m = 512$ MIA still reaches a TPR@5%FPR of 0.492–0.562, about ten times the false-positive rate.

Output granularity. So far, all results are reported for APIs that return a full confidence vector. For regression, this is the bar distribution or a set of quantiles, both of which are offered in practice. The hosted TabPFN API returns either one (Prior Labs, 2026), and forecasting services return quantiles at arbitrary levels (Google Cloud, 2026a; Ansari et al., 2025). Other services return less. SAP-RPT-1.5 returns a point and an interval (SAP, 2026), TabFM in BigQuery returns a point (Google Cloud, 2026b), and SageMaker Autopilot endpoints, by default, return only the predicted label (Amazon Web Services, 2026).

Table 3 reports each attack under these formats; §B.5 describes how the attacks are adapted to them. On `students_dropout`, returning the top-1 label with its confidence leaves every attack close to its full-vector value. When only the label is returned, the stolen surrogate stays faithful (fidelity 0.855 against 0.874), but the privacy attacks no longer beat their baselines. On the regression dataset `houses`, returning only the point estimate has little effect: only AIA drops, from 0.685 to 0.285, and it still exceeds its baseline of 0.130. Restricting the output can thus mitigate the privacy risk for classification, but not for regression.

5 CONCLUSION

In this work, we showed that deployed TFMs expose their in-context data to a considerably greater extent than black-box access suggests. An adversary who queries the deployed model with randomly generated rows, without any in-distribution data and without write access to the context, can obtain a stolen surrogate with high fidelity to the target model. This surrogate can then be used to run privacy attacks locally and without any restrictions. We adapted membership inference, attribute inference, and property inference to this setting and found that all three leak significantly more private information than when run against the deployed model directly, in some cases approaching the oracle baseline that requires much stronger access. The approach remains effective when the adversary does not know the backbone or the context size of the target model, and the leakage increases with task difficulty. Restricting the output to the predicted label prevents the privacy attacks for classification, but not stealing itself, and regression outputs still leak. Our results suggest that providers who serve TFMs adapted on sensitive data should not treat black-box access as a sufficient privacy protection, and motivate the development of dedicated defenses for in-context data.

AI USE STATEMENT

In this work, we used generative AI tools to implement parts of the attack and evaluation code. AI-assisted code and data preprocessing were reviewed and tested by the authors, and all reported numbers are produced by our own experiment scripts. AI-assisted interpretations of results were checked by the authors against the experimental outputs. Given the experiment results, we used AI to generate the figures. Additionally, we used generative AI tools to search for, identify, and summarize related literature, and for a final check of reference formatting; all citations were selected and added by the authors. All AI-drafted or AI-edited text was reviewed and revised by the authors, and AI-identified literature was checked against the original publications. We have not used AI for tasks outside of the aforementioned ones. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Amazon Web Services. Configure inference output in Autopilot-generated containers, 2026. URL https://docs.aws.amazon.com/sagemaker/latest/dg/autopilot-automate-model-development-container-output.html. Accessed 2026-09-25.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting. *CoRR*, abs/2510.15821, 2025. doi: 10.48550/ARXIV.2510.15821. URL https://doi.org/10.48550/arXiv.2510.15821.

Giuseppe Ateniese, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *Int. J. Secur. Networks*, 10(3):137–150, 2015. doi: 10.1504/IJSN.2015.071829. URL https://doi.org/10.1504/IJSN.2015.071829.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pp. 1897–1914. IEEE, 2022. doi: 10.1109/SP46214.2022.9833649. URL https://doi.org/10.1109/SP46214.2022.9833649.

Nicholas Carlini, Daniel Paleka, Krishnamurthy Dj Dvijotham, Thomas Steinke, Jonathan Hayase, A. Feder Cooper, Katherine Lee, Matthew Jagielski, Milad Nasr, Arthur Conmy, Eric Wallace, David Rolnick, and Florian Tramèr. Stealing part of a production language model. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 5680–5705. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/carlini24a.html.

Tânia Carvalho and Maxime Cordy. Privacy vulnerabilities of attention layers in tabular foundation models and protection of high-risk queries. *CoRR*, abs/2606.26021, 2026. doi: 10.48550/ARXIV.2606.26021. URL https://doi.org/10.48550/arXiv.2606.26021.

Christopher A. Choquette-Choo, Florian Tramèr, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1964–1974. PMLR, 2021. URL http://proceedings.mlr.press/v139/choquette-choo21a.html.

Clarifai. Clarifai. https://www.clarifai.com/. Accessed: 2026-09-26.

Cohere. Cohere. https://cohere.ai/. Accessed: 2026-09-26.

Moonjung Eo, Min-Kook Suh, Hye-Seung Cho, Jiwon Kim, Seoyoon Kim, Sangjun Nam, and Soonyoung Lee. EXAONE tabular 1.0 : Technical report. *CoRR*, abs/2608.25774, 2026. doi: 10.48550/ARXIV.2608.25774. URL https://doi.org/10.48550/arXiv.2608.25774.

Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Mutalik Desai, David Salinas, and Frank Hutter. Tabarena: A living benchmark for machine learning on tabular data. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/1697e3fb412da11dc9488249f9e7bbc9-Abstract-Datasets_and_Benchmarks_Track.html.

Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon M. Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In Kevin Fu and Jaeyeon Jung (eds.), *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014*, pp. 17–32. USENIX Association, 2014. URL https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/fredrikson_matthew.

Karan Ganju, Qi Wang, Wei Yang, Carl A. Gunter, and Nikita Borisov. Property inference attacks on fully connected neural networks using permutation invariant representations. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang (eds.), *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pp. 619–633. ACM, 2018. doi: 10.1145/3243734.3243834. URL https://doi.org/10.1145/3243734.3243834.

Google Cloud. Online inference for forecasting, Gemini Enterprise Agent Platform (formerly Vertex AI), 2026a. URL https://cloud.google.com/vertex-ai/docs/tabular-data/tabular-workflows/forecasting-online-predictions. Accessed 2026-09-25.

Google Cloud. The AI.PREDICT function, BigQuery documentation, 2026b. URL https://docs.cloud.google.com/bigquery/docs/reference/standard-sql/bigqueryml-syntax-ai-predict. Preview feature. Accessed 2026-09-25.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Benjamin Jäger, Dominik Safaric, Simone Alessi, Adrian Hayler, Mihir Manium, Rosen Yu, Felix Jablonski, Shi Bin Hoo, Anurag Garg, Jake Robertson, Magnus Bühler, Vladyslav Moroshan, Lennart Purucker, Clara Cornu, Lilly Charlotte Wehrhahn, Alessandro Bonetto, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. Tabpfn-2.5: Advancing the state of the art in tabular foundation models. *CoRR*, abs/2511.08667, 2025. doi: 10.48550/ARXIV.2511.08667. URL https://doi.org/10.48550/arXiv.2511.08667.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, Dominik Safaric, Jake Robertson, Benjamin Jäger, Simone Alessi, Adrian Hayler, Vladyslav Moroshan, Lennart Purucker, Philipp Singer, Alan Arazi, Julien Siems, Jan Hendrik Metzen, Georg Grab, Nick Erickson, Siyuan Guo, Eliott Kalfon, Simon Bing, David Salinas, Clara Cornu, Lilly Charlotte Wehrhahn, Diana Kriuchkova, Kursat Kaya, Lydia Sidhoum, Marie Salmon, Jerry Chen, Madelon Hulsebos, Yann LeCun, Samuel Müller, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. Tabpfn-3: Technical report. *CoRR*, abs/2605.13986, 2026. doi: 10.48550/ARXIV.2605.13986. URL https://doi.org/10.48550/arXiv.2605.13986.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular

foundation model. *Nat.*, 637(8045):319–326, 2025. doi: 10.1038/S41586-024-08328-6. URL https://doi.org/10.1038/s41586-024-08328-6.

Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. High accuracy and high fidelity extraction of neural networks. In Srdjan Capkun and Franziska Roesner (eds.), *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pp. 1345–1362. USENIX Association, 2020. URL https://www.usenix.org/conference/usenixsecurity20/presentation/jagielski.

Joonas Jälkö, Gauri Pradhan, Ossi Räisä, and Antti Honkela. On reliability of efficient membership inference vulnerability evaluation. *CoRR*, abs/2605.25819, 2026. doi: 10.48550/ARXIV.2605.25819. URL https://doi.org/10.48550/arXiv.2605.25819.

Weihao Kong and Abhimanyu Das. Introducing TabFM: A zero-shot foundation model for tabular data. Google Research Blog, 2026. Model and code at https://github.com/google-research/tabfm; no technical report released at time of writing.

Kalpesh Krishna, Gaurav Singh Tomar, Ankur P. Parikh, Nicolas Papernot, and Mohit Iyyer. Thieves on sesame street! model extraction of bert-based apis. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Byl5NREFDr.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Alex Labach, Jesse C. Cresswell, Keyvan Golestan, Guangwei Yu, Anthony L. Caterini, and Maksims Volkovs. Tabdpt: Scaling tabular foundation models on real data. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/fc0e3f908a2116ba529ad0a1530a3675-Abstract-Conference.html.

Samuel Müller, Noah Hollmann, Sebastian Pineda-Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

OpenAI. OpenAI. https://openai.com/. Accessed: 2026-09-26.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 4949–4958. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00509. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Orekondy_Knockoff_Nets_Stealing_Functionality_of_Black-Box_Models_CVPR_2019_paper.html.

Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In Ramesh Karri, Ozgur Sinanoglu, Ahmad-Reza Sadeghi, and Xun Yi (eds.), *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, pp. 506–519. ACM, 2017. doi: 10.1145/3052973.3053009. URL https://doi.org/10.1145/3052973.3053009.

Prior Labs. TabPFN API: Getting started, 2026. URL https://docs.priorlabs.ai/api-reference/getting-started. Accessed 2026-09-25.

PriorLabs. Tabpfn capabilities: Embeddings. https://docs.priorlabs.ai/capabilities/embeddings, 2025.

Prottay Protivash, John Durrell, Daniel Kifer, Zeyu Ding, and Danfeng Zhang. Reconstruction attacks on aggressive relaxations of differential privacy. *J. Priv. Confidentiality*, 14(3), 2024. doi: 10.29012/JPC.871. URL https://doi.org/10.29012/jpc.871.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabicl: A tabular foundation model for in-context learning on large data. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/qu25d.html.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabiclv2: A better, faster, scalable, and open tabular foundation model. *CoRR*, abs/2602.11139, 2026. doi: 10.48550/ARXIV.2602.11139. URL https://doi.org/10.48550/arXiv.2602.11139.

SAP. SAP-RPT-1.5: A step-by-step guide on getting started, 2026. URL https://community.sap.com/t5/artificial-intelligence-blogs-posts/sap-rpt-1-5-a-step-by-step-guide-on-getting-started/ba-p/14290171. Accessed 2026-09-25.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pp. 3–18. IEEE Computer Society, 2017. doi: 10.1109/SP.2017.41. URL https://doi.org/10.1109/SP.2017.41.

Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of machine learning models. In Michael D. Bailey and Rachel Greenstadt (eds.), *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pp. 2615–2632. USENIX Association, 2021. URL https://www.usenix.org/conference/usenixsecurity21/presentation/song.

Anshuman Suri and David Evans. Formalizing and estimating distribution inference risks. *Proc. Priv. Enhancing Technol.*, 2022(4):528–551, 2022. doi: 10.56553/POPETS-2022-0121. URL https://doi.org/10.56553/popets-2022-0121.

Yefan Tao, Xiyuan Zhang, Xinyi Liu, Boran Han, Danielle Maddix, Haoyang Fang, Zhen Han, Jiading Gai, Xuanqing Liu, Michael Bohlke-Schneider, Yuyang Wang, Gerald Friedland, Kevan Mah, Chris Lee, and Chris Kong. Mitra-v2 technical report. 2026. URL https://arxiv.org/abs/2609.04540.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In Thorsten Holz and Stefan Savage (eds.), *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016*, pp. 601–618. USENIX Association, 2016. URL https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/tramer.

Florian Tramèr, Reza Shokri, Ayrton San Joaquin, Hoang Le, Matthew Jagielski, Sanghyun Hong, and Nicholas Carlini. Truth serum: Poisoning machine learning models to reveal their secrets. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022*, pp. 2779–2792. ACM, 2022. doi: 10.1145/3548606.3560554.

Dariush Wahdany, Matthew Jagielski, Jesse C. Cresswell, Adam Dziedzic, and Franziska Boenisch. Tabpate: Differentially private tabular in-context learning without public data. *CoRR*, abs/2606.31474, 2026. doi: 10.48550/ARXIV.2606.31474. URL https://doi.org/10.48550/arXiv.2606.31474.

Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi (eds.), *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pp. 3093–3106. ACM, 2022. doi: 10.1145/3548606.3560675. URL https://doi.org/10.1145/3548606.3560675.

Chin-Chia Michael Yeh, Uday Singh Saini, Xin Dai, Xiran Fan, Shubham Jain, Yujie Fan, Jiarui Sun, Junpeng Wang, Menghai Pan, Yingtong Dou, Yuzhong Chen, Vineeth Rakesh, Liang Wang, Yan Zheng, and Mahashweta Das. Treasure: The visa payment foundation model for high-volume transaction understanding, 2025. URL https://arxiv.org/abs/2511.19693.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018*, pp. 268–282. IEEE Computer Society, 2018. doi: 10.1109/CSF.2018.00027. URL https://doi.org/10.1109/CSF.2018.00027.

Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. Low-cost high-power membership inference attacks. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 58244–58282. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/zarifzadeh24a.html.

Wanrong Zhang, Shruti Tople, and Olga Ohrimenko. Leakage of dataset properties in multi-party machine learning. In Michael D. Bailey and Rachel Greenstadt (eds.), *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pp. 2687–2704. USENIX Association, 2021. URL https://www.usenix.org/conference/usenixsecurity21/presentation/zhang-wanrong.

A EXPERIMENTAL SETUP

Table 4 collects the default parameters used for the experiments, with additional details presented in the paragraphs below. Unless a figure or table says otherwise, every number in §4 is measured in this configuration.

Table 4: Default configuration of the main experiments.

Service	
Victim backbone θ	TabICL-v2 (Qu et al., 2026), one ensemble member, seed 0
Context size m	256 records of the table
Output	full confidence vector (classification); 10-cell vector (regression)
Adversary	
Public model $\tilde{\theta}$	TabICL-v2, one ensemble member, seed 1
Surrogate prompt size n	256
Transfer budget B	$2^{16} = 65{,}536$ victim queries
Surrogate copies K	32, shared by all four attacks
Fitted parts	context tokens t and weights $\tilde{\theta}$, jointly
Optimizer	AdamW, no weight decay, lr 10^{-2} (t or R) / 10^{-3} ($\tilde{\theta}$)
Batch / epochs	1024 rows / at most 100 epochs over D_T, early stopping after 10
MIA refinement	on: each copy refined on a Hamming ball of $B/2 = 32{,}768$ rows around z
Evaluation	
Held-out set	$\min(1{,}024,\ \lvert D\rvert - 256 - 128)$ records
MIA pool	128 members + 128 non-members
AIA targets	100 context members per table, $\lvert U\rvert \in \{1, \ldots, 5\}$
PIA probes	$N_R = 2{,}048$ per region, top 10% by RANK kept

Models and serving. The victim and the surrogate both run the public TabICL-v2 model (`tabicl` 2.2.0), with a single ensemble member rather than the upstream default of 8. A TabICL ensemble member is not a separately trained model but one preprocessing view of the same context, and averaging views adds a nuisance source of variance and multiplies the cost of every query. Numbers measured at one member therefore describe a single-view service, and the deployed default is generally more accurate. Victim and surrogate use different random seeds (0 and 1), so the adversary never holds the victim's preprocessing draw. Both receive the typed table: one column per attribute with its declared categories, with categorical handling left to the model rather than to a one-hot expansion of ours. For the generality study (§4.2), the victim and the surrogate each range over the public TabICL-v2, RealTabPFN-2.5 (Grinsztajn et al., 2025) and TabPFN-3 (Grinsztajn et al., 2026) checkpoints, and m and n each over $\{64, 128, 256, 512\}$, with every other setting held at its default.

Datasets and splits. We load each TabArena table from OpenML, drop rows with missing values, and subsample tables larger than 20,000 rows to 20,000. One random permutation per table (seed 0) is sliced into the served context D_{ctx} (the first $m = 256$ rows), a held-out evaluation set of $\min(1{,}024,\ \lvert D\rvert - 256 - 128)$ rows, and 128 non-members for MIA, with the three sets being disjoint. The smallest table, `blood_transfusion`, has 748 rows, and thus a held-out set of 364. The declared schema the adversary holds (column types, categories and numeric ranges) is read from the full table, as a benign user of the service would see it.

Regression. For the 12 regression tables the victim is the TabICL-v2 regressor, conditioned on the real targets of its 256 context records. Its answer is read as a 10-cell vector over the declared target range, with the two outermost cells open-ended, as described in §B.4. TabICL-v2 answers in 999 quantiles, which we interpolate into the cells with a monotone cubic (PCHIP) spline. The labels of the regression attacks are these cells: MIA and AIA read the cell holding the record's target, and PIA estimates the cell composition of a region. The surrogate is the public TabICL-v2 classifier, fitted to the 10-cell vectors exactly as to a classifier's posterior.

Transfer set and surrogate copies. D_T is built in two halves of $B/2$ rows each. The first is drawn uniformly from the declared schema. The second is generated by local perturbation, starting from the first-half rows to which the victim assigns the highest probability of each class that the first half leaves under-represented, so that the transfer set is not dominated by the victim's majority class. If no class is under-represented, the second half is also uniform. All B rows are distinct, and on a

table whose whole domain holds fewer than B legal rows, D_T is the domain itself, so the victim is queried once per distinct row. Each copy starts from $n = 256$ rows of D_T drawn at random with the copy's own seed, after which every class the victim predicts on D_T is forced into the prompt at least once (the *mandatory* rule of §B.6). The copy is then fitted by minimizing the cross-entropy from the victim's recorded vector on D_T to its own, updating the prompt's token block t and the weights $\tilde{\theta}$ jointly, each with its own learning rate and gradient-norm clip at 1.0. Fitting stops after 10 consecutive epochs without improvement in the training loss, or after 100 epochs, and restores the best iterate. Because the $K = 32$ copies differ only in their initial prompt and optimizer seed, the victim is queried once for all of them, and the same 32 copies serve as the MIA shadows, the AIA reference and the PIA committee. One consequence is that the four attacks on a table are not independent observations: a copy set that happens to be good or bad is shared by all four.

Functionality stealing. The surrogate is scored on the held-out set and on the 256 context members separately. Per table, each metric is averaged over the surrogates. Figure 2 reports the mean over tables. In Figure 2(a), each fitted part is run with 4 copies per table and the same prompt, seed, loss and early-stopping rule, so that only the free parameters differ. In Figure 2(b), a single copy per table is fitted at each $B \in \{2^8, 2^{10}, \ldots, 2^{20}\}$. At $B = 2^8$, 11 of the 33 classification tables yield a transfer set with a single class and are omitted.

Membership inference. The 128 members are drawn from D_{ctx} and the 128 non-members from the table outside it, so both come from the same distribution. With 128 non-members, the 1% FPR operating point rests on a single false positive and TPR moves in steps of $1/128$; we therefore also report TPR@5%FPR where noted. For the per-record refinement, the Hamming ball around each candidate holds $B/2 = 32{,}768$ rows. The ORACLE is a shadow-model LiRA run on the victim's backbone: each shadow context holds $m = 256$ records, half of them candidates and half real filler rows of the same table (violating the threat model), and the shadows come in complementary pairs, so every candidate is IN one shadow of a pair and OUT of the other. Figure 5 subsamples $K \in \{1, 2, 4, \ldots, 32\}$ copies from the 32 fitted ones, 5 subsets per K.

Attribute inference. For each target, $|U|$ unknown columns are drawn at random, and the remaining columns together with the label form the anchor. Categorical columns take their declared categories as candidates, while numeric columns are discretized onto a 16-point grid over their declared range. A numeric cell counts as recovered when the chosen value falls in the same grid cell as the truth. The candidate set is the product of the unknown columns' candidates, sampled down to 256 per target when it is larger. The share of targets whose true record survives this cap is reported as coverage (COV. in Table 2a), and it bounds the record-level accuracy of every method. Results are averaged with equal weight over the 45 tables.

Property inference. For each number of conditions $\ell \in \{1, 2, 3\}$, we take up to 3 regions, plus the whole context ($\ell = 0$). The regions are the conjunctions of categorical conditions that occur most often in D_{ctx}, among those that hold for between 12 and 96 of its 256 records; without the upper limit, the most frequent regions cover nearly the whole context and all share its label distribution. This selection reads the private context, which the threat model does not grant, so for $\ell \geq 1$ the results measure how well the attack recovers α on a given region, not how well an adversary could find that region. The whole context ($\ell = 0$) involves no selection and is unaffected. Since regions need categorical columns, a table without any contributes only $\ell = 0$: $\ell = 0$ to 3 are averaged over 45, 29, 26 and 24 tables, respectively. Each region is probed with $N_R = 2{,}048$ schema-uniform records, and the NON-STEALING baseline reports the target model's pseudo-label composition of all of them, without pruning.

Compute. All experiments run on single NVIDIA A40, L40 or A100 (40 GB) GPUs, with PyTorch 2.11. The cost is dominated by the MIA refinement balls, $32{,}768 \times 256$ victim-labeled rows per table. Victim queries are counted once per distinct row, so a row queried by one attack and read again by another counts once.

Table 5: **Comparison between refined and unrefined copies of the surrogate.** The surrogate copies in MIA can be refined around the target record or not. Numbers are averaged over the datasets with $K = 32$ copies.

Method	Queries / cand.	TPR@5%FPR	AUC
Unrefined copies	1	0.340	0.730
Refined copies	32, 769	0.359	0.750
BASELINE	1	0.217	0.705
ORACLE	1	0.540	0.875

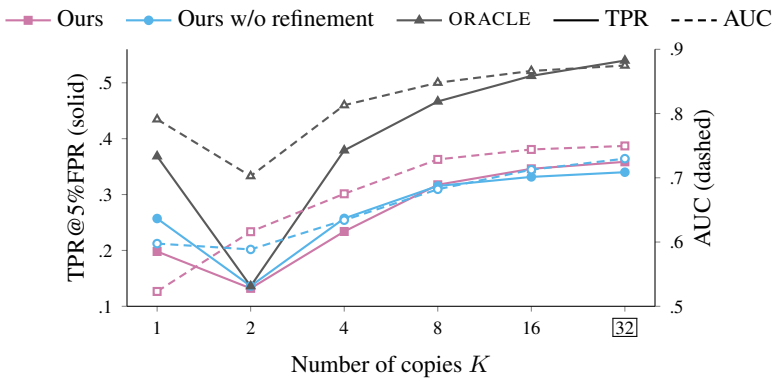


Figure 5: **MIA: how many surrogate copies the attack needs.** Each point is the mean over the datasets. Solid lines (left axis): true-positive rate at a 5% false-positive rate; dashed lines (right axis): area under the ROC curve, where 0.5 is random guessing. *Ours* is the MIA attack with the per-record refinement of §3.2, and *Ours w/o refinement* skips that step. The ORACLE is a standard shadow-model attack (LiRA) whose shadow models are built from real records of the victim's data distribution. It violates the threat model and serves as an upper reference, not as an attack; for it, K is the number of shadow model pairs. All copies are fitted to the same victim answers, so a larger K costs attacker compute but no additional victim queries. The boxed tick $K = 32$ is used in all other experiments.

B Detailed Analysis of Attacks

B.1 Per-record refinement for membership inference

Table 5 turns the per-record refinement of §3.2 on and off. The ball contains every Hamming-1 neighbor of the challenge record, topped up with sampled Hamming-2 rows until it holds $B/2$ of them, with the record itself and the transfer set excluded. The ball is generated from z, so the refinement stays within the threat model. The victim labels it once, and all K copies share that labeling, so the ball is bought once rather than once per copy. Each copy is then trained for a single epoch over the ball, with the weights frozen so that only the fitted context moves. Refining is what separates a member from its neighborhood rather than from the population, and the ensemble's spread after it measures local instead of global uncertainty. It is also the expensive half of the attack: the unrefined method spends one victim query per candidate, and the refined one spends 32,769, the ball plus the candidate itself. With $K = 32$ copies this buys 0.019 in TPR@5%FPR (0.340 to 0.359) and 0.020 in AUC (0.730 to 0.750), against 0.540 and 0.875 for the ORACLE (Table 5). The gain depends on the number of copies (Figure 5): with a single copy the refined method is worse on both metrics, and it leads on AUC from $K = 2$ and on TPR@5%FPR only from $K = 8$.

B.2 Number of surrogate copies for membership inference

Figure 5 varies the number of surrogate copies K that MIA reads, subsampling K of the 32 fitted copies with 5 subsets per K (§A). With per-record refinement, AUC rises from 0.52 at $K = 1$ to 0.75 at $K = 32$ and TPR@5%FPR from 0.20 to 0.36; both change little beyond $K = 16$.

B.3 THE ORACLE FOR ATTRIBUTE INFERENCE

The ORACLE adapts Truth Serum (Tramèr et al., 2022) to attribute inference. For a target z with label y_z and a candidate completion x from its candidate set, let y' be the label the unpoisoned victim ranks highest at x after y_z. We append r copies of (x, y') to D_{ctx}, serve the victim on the enlarged context, and read its confidence in y_z at x as a clipped logit $s_r(x)$, for $r \in \{1, 2, 4, 8\}$. On a regression table served as a confidence vector (§B.4), y' is a cell of the vector, and each decoy carries the target value at that cell's center. The candidate is scored by $\frac{1}{4}\sum_r s_r(x)$, and the highest-scoring candidate is emitted. The true completion is already a record of D_{ctx} with label y_z, so the decoys must outvote it and the confidence in y_z stays high; a wrong completion has no such record, so the same decoys pull its confidence in y_z further down. The score reads the confidence *after* poisoning rather than how far it drops: a candidate the victim already rates low on y_z has little confidence left to lose, and a drop-based score mistakes that for resistance to the decoys. Every decoy is served through the victim's own model with its preprocessing refitted, rather than approximated. The method needs 4 served contexts per candidate, so it is evaluated on the first 10 of the targets per table and $|U|$; on those targets, scoring each candidate by the unpoisoned victim's confidence in y_z gives a per-cell accuracy within 0.03 of its value over all targets, at every $|U|$.

B.4 READING A REGRESSION SERVICE AS A CONFIDENCE VECTOR

A regressor does not return a confidence vector, so the attacker builds one from whatever the head natively answers. One readout is fixed per head family, and both are indexed by the public schema rather than by anything the served context decides.

A head carrying a density or an exact CDF is read as bin masses: $P(y \in b)$ over g bins tiling the declared range $[lo, hi]$, plus the two open tails $(-\infty, lo)$ and (hi, ∞) so that no belief is discarded. The row sums to one and has the shape of a class posterior, which is what lets the attacks of §3 read a regression service unchanged.

The edges have to come from the schema rather than from the model. TabPFN answers in bars whose borders are refit per context: a fixed confidence vector rescaled by the served targets' own mean and standard deviation. Two contexts are therefore answered on two different indexings, and their vectors cannot be compared cell by cell. Binning on the public edges keeps the values, which carry the signal, and drops only that private indexing. TabICL answers in quantiles. To reach the same output as TabPFN, we interpolate the quantiles into the same public bins.

B.5 MODIFYING THE ATTACK ON DIFFERENT OUTPUT FORMATS

A service provider can decide to restrict this output to limit the information that is exposed to the customer. We define three output formats for classification and two for regression.

For classification, the default is exposing the entire confidence vector. The first restriction is to expose only the top label and its corresponding confidence score. An additional restriction would be to expose only the predicted label. Similarly, for regression, the default is to expose a full distribution, and the restriction is to expose only the output value.

We modify our attacks for these output modes in the following way.

Top-1 label and confidence. The adversary completes each answer to a full confidence vector: the returned label keeps its confidence, and the remaining mass is split equally over the other classes. With two classes, this recovers the vector exactly. All four attacks then run unchanged on the completed vectors. Only the target model's answers are completed; the surrogates still output full vectors.

Top-1 label. For STEALING, the transfer set is built entirely with uniform-random queries. It fits the surrogates to one-hot targets of the returned labels. The other three attacks have no confidence to read. They use instead how stable a label is under perturbation: a record in the context is answered by attending to itself, so its label should change less when the record is perturbed. A perturbation adds Gaussian noise with 0.1 times the column's standard deviation in the transfer set to every numerical attribute and changes each categorical attribute to a different value with probability 0.1. The stability of a record is the fraction of its perturbations that receive the same label as the record itself. A record

far inside one class is stable under any model, member or not, so every score subtracts the mean stability of the K surrogates on the same perturbations. MIA scores the candidate z this way, with 50 perturbations. AIA scores a candidate completion by the fraction of it and its neighbors, which change only the known attributes, that the target model labels y_z, minus the same fraction for the surrogates. PIA replaces RANK (Eq. (4)) with this calibrated stability, using 16 perturbations per probe, and keeps the top-ranked probes as before.

Point estimate. The surrogates are copies of the public TabICL-v2 regressor rather than the classifier used on the 10-cell vectors (§A). STEALING fits them to the target model's point answers with an L1 loss divided by the width $hi - lo$ of the declared target range. The surrogate's point answer is computed as the target model computes its own, as the mean of the predicted quantiles. We fit the weights, the context, or both, and keep the part with the lowest training loss on the transfer set, which is the weights. The other attacks measure how close an answer $f(x)$ lands to a value y by

$$s(x; y) = -\log\left(\frac{|f(x) - y|}{hi - lo} + 10^{-6}\right). \tag{5}$$

A single closeness is dominated by how easy the region is to predict, so each attack uses the local contrast: $s(x; y)$ minus its mean over perturbations of x (same noise as above), again minus the same quantity on the surrogates, averaged over the K surrogates. MIA uses $x = z$ and $y = y_z$ with 50 perturbations. AIA scores each candidate completion with $y = y_z$, perturbing only its known attributes, with 16 perturbations. PIA estimates the share of the context's targets in each of the 10 cells: it scores each probe with $y = f_V(x)$, the target model's own answer, over 16 perturbations, and reports the histogram of the target model's answers on the probes with high scores.

B.6 FALLBACK METHODS FOR SINGLE CLASS COLLAPSE

The transfer set is created by querying the victim model with random rows following the known schema, and labeling them with the victim's output. We have observed that, for some tables, the same label is assigned to the majority of rows, leading to each row in a 256 row sample having the same label. We refer to this as single class collapse.

Since this collapse renders the sample unusable, we have added two fallback methods to apply in this case. The *resample* method takes the top 256 rows of the minority class, and appends their neighbors to the transfer set, labeled with that same class. The *mandatory* method enforces the addition of at least one row of each class contained in the transfer set into the sample.