Iterative Subset Selection for High-fidelity Synthetic Tabular Data

Daniel Gärber
Independent Researcher
danielgaerber8@gmail.com

Lea Demelius
Know Center Research GmbH
University of Technology, Graz
ldemelius@know-center.at

Abstract

We present ISSOSYNTH, an iterative subset selection method for generating high-fidelity synthetic tabular data. The approach won the Mostly AI prize (2025) for generating the synthetic data with the highest fidelity. We evaluate the fidelity, utility and empirical privacy of the approach on various datasets and show that the method is able to improve fidelity and utility on downstream tasks without notably increasing vulnerability to membership inference attacks.

1 Introduction

Many AI applications rely on high-quality tabular datasets, especially in fields such as healthcare, finance, and industry. However, real-world data sharing is often restricted by privacy and confidentiality concerns, making synthetic data generation a practical and widely adopted alternative. Synthetic data mimics the statistical properties of real data, while eliminating the direct link to individual records. Although numerous synthetic data generators (SDGs) have emerged, ranging from statistical approaches to deep generative models, ensuring that the resulting synthetic datasets maintain high fidelity and utility for downstream tasks remains a challenge.

In this work, we present **I**terative **S**ubset **S**election from **O**versampled **SYNTH**etic data (ISSOSYNTH), a generator-agnostic method to improve the fidelity of synthetic tabular data. It achieves this by oversampling any SDG and iteratively selecting a subset that best preserves the underlying data distribution. ISSOSYNTH won the Mostly AI Prize (2025)[1], a global synthetic tabular data competition taking place earlier this year. We evaluate the approach on multiple datasets and SDGs. To provide a comprehensive assessment, we consider not only data fidelity but also the utility on downstream classification tasks and estimate privacy leakage empirically using state-of-the-art membership inference attacks.

2 Method

2.1 ISSOSYNTH: Iterative Subset Selection from Oversampled SYNTHetic data

To enhance the fidelity of generated data, we propose the multi-stage selection algorithm ISSOSYNTH that filters a large, oversampled synthetic data pool to produce a high-fidelity subset. ISSOSYNTH is designed to closely align the statistical properties of the synthetic data with the real-world source data by sequentially applying three distinct procedures:

1. **Iterative Proportional Fitting (IPF)** for initial candidate selection

[1] https://www.mostlyaiprize.com/

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: AI for Tabular Data workshop at EurIPS 2025.

2. **Greedy Trimming** phase to fine-tune while reducing the set to the target size
3. **Refinement** stage to fine-tune the final dataset with samples from the synthetic data pool

The approach requires training a generative model and using it to create a synthetic data pool significantly larger than the source dataset. This oversampling ensures a diverse set of candidates for the selection process. The complete process is detailed in Algorithm 1. For clarity, the pseudocode presents a simplified version of the core logic; further implementation details are discussed in the appendix. It takes the real source dataset D_{real} and a larger, oversampled synthetic pool D_{pool} (where $|D_{\text{pool}}| > |D_{\text{real}}|$) as input. It aims to select a final subset D_{final} of target size N (we use $N = |D_{\text{real}}|$). Our core optimization metric, $\mathcal{L}_1(D_A, D_B)$, represents the total, normalized L_1 distance over all monitored uni-, bi-, and trivariate distributions between dataset D_A and D_B. This metric was specifically chosen as it was the target metric of the Mostly AI competition. To efficiently compute this, all features are discretized into 10 bins, and the metric is calculated on the binned representations.

Subset Selection via Iterative Proportional Fitting (IPF) The objective of the first stage, IPF-SELECTION, is to select an initial, oversized subset D_{sample} from the synthetic data pool D_{pool} that closely mirrors the bivariate distributions of the real data D_{real}. To manage computational complexity, the algorithm first selects the top k_{pairs} feature pairs P based on mutual information. It then iteratively adjusts a weight vector w for all samples in D_{pool}. In each iteration, it creates the target contingency tables T_p from D_{real} and the current weighted contingency tables C_p from D_{pool} for each pair $p \in P$. Scaling factors S_p are calculated and used to update the sample weights w. This process repeats for K_1 iterations. Finally, an oversized subset D_{sample} of size N_{sample} is sampled from D_{pool} according to these final weights.

Greedy Trimming We employ a greedy trimming algorithm to reduce the oversized subset from the IPF phase down to the final target size N. This is achieved by iteratively identifying and removing a batch of S_{trim} data points X^- from the current set D_{curr}. The samples to be removed are selected based on having the highest removal gain $g(x)$, meaning their removal most improves the total $\mathcal{L}_1$ distance relative to D_{real}. This batch removal process is repeated until the dataset reaches the target size N.

Refinement via Swapping Finally, the trimmed dataset undergoes an iterative refinement process to further enhance its fidelity. In each iteration, the algorithm first identifies a batch of S_{swap} worst-fitting records X^- within the current subset, i.e., the records with the highest $g_{\text{remove}}(x)$. It then searches the oversampled pool (excluding those already in the subset) for the best-fitting batch of S_{swap} candidates X^+ to add in their place.

2.2 Evaluation

We evaluate our method across three axes: **fidelity**, **utility**, and **privacy**. We test our generator-agnostic method, ISSOSYNTH, on four popular generators: Gaussian Copula, CTGAN [13], TVAE [13], and TabularARGN [9]. We use four openly accessible tabular datasets: Adult [1], Bank Marketing [4], Electricity [2], and Nursery [7]. For each experiment, we split the original dataset into three equally sized, disjoint parts: a **training** set, a **test** set, and a **reference** set. The synthetic data generators (SDGs) are trained only on the training set. We create two synthetic datasets from this: a baseline synthetic dataset without any post-processing, and our optimized dataset using ISSOSYNTH. This entire process is repeated three times with different random splits to ensure robust results.

We measure fidelity using the average of the three accuracy scores for univariate, bivariate, and trivariate distributions, as implemented in the Mostly AI data benchmark implementation [8]. We compare the baseline and optimized synthetic datasets against the training set, and report the fidelity difference in Table 1 as *fidelity gain*.

To evaluate downstream utility, we train a gradient boosting model once on the baseline synthetic dataset and once on the dataset optimized with ISSOSYNTH. We then measure its predictive performance on the test set using the Area Under the Receiver Operating Characteristic Curve (AUC-ROC), and report the difference between the AUC-ROC as *utility gain* in Table 1.

Algorithm 1 ISSOSYNTH

Input: real dataset D_{real}, synthetic data pool from SDG D_{pool}
Output: Optimized subset D_{final}
$D_{\text{sample}} \leftarrow \text{IPFSELECTION}(D_{\text{real}}, D_{\text{pool}})$
$D_{\text{curr}} \leftarrow \text{GREEDYTRIMMING}(D_{\text{real}}, D_{\text{sample}})$
$D_{\text{final}} \leftarrow \text{REFINEMENT}(D_{\text{real}}, D_{\text{pool}}, D_{\text{curr}})$

function IPFSELECTION(D_{real}, D_{pool})
    $P \leftarrow$ top k_{pairs} feature pairs from D_{real} by mutual information
    Initialize weights $w_i \leftarrow 1$ for each $x_i \in D_{\text{pool}}$
    for $i = 1$ to K_1 **do** ▷ K_1: max iterations
        for each pair $p \in P$ **do**
            $T_p \leftarrow$ contingency table of feature pair p on binned D_{real}
            $C_p \leftarrow$ contingency table of feature pair p on binned D_{pool} weighted by w
            $S_p \leftarrow T_p / C_p$ ▷ Calculate cell-wise scaling factors
            $B_p \leftarrow$ lookup array for each $x_i \in D_{\text{pool}}$ to its bin index in S_p
            $w \leftarrow w \times S_p[B_p]$
    $D_{\text{sample}} \leftarrow$ sample N_{sample} from D_{pool} with weights w ▷ N_{sample}: intermediate dataset size
    return D_{sample}

function GREEDYTRIMMING(D_{real}, D_{curr})
    while $|D_{\text{curr}}| > N$ **do** ▷ N: target dataset size
        For $x \in D_{\text{curr}}$, calculate gain $g(x) = \mathcal{L}_1(D_{\text{real}}, D_{\text{curr}}) - \mathcal{L}_1(D_{\text{real}}, D_{\text{curr}} \setminus \{x\})$
        $X^- \leftarrow S_{\text{trim}}$ samples from D_{curr} with the highest gain $g(x)$ ▷ S_{trim}: trim batch size
        $D_{\text{curr}} \leftarrow D_{\text{curr}} \setminus \{X^-\}$
    return D_{curr}

function REFINEMENT(D_{real}, D_{pool}, D_{curr})
    for $i = 1$ to K_2 **do** ▷ K_2: max iterations
        For $x \in D_{\text{curr}}$, calculate $g_{\text{remove}}(x) = \mathcal{L}_1(D_{\text{real}}, D_{\text{curr}}) - \mathcal{L}_1(D_{\text{real}}, D_{\text{curr}} \setminus \{x\})$
        $X^- \leftarrow S_{\text{swap}}$ samples from D_{curr} with the highest $g_{\text{remove}}(x)$ ▷ S_{swap}: swap batch size
        $D' \leftarrow D_{\text{curr}} \setminus \{X^-\}$
        $D_{\text{pool\_remaining}} \leftarrow D_{\text{pool}} \setminus D_{\text{curr}}$
        For $x' \in D_{\text{pool\_remaining}}$, calculate $g_{\text{add}}(x') = \mathcal{L}_1(D_{\text{real}}, D') - \mathcal{L}_1(D_{\text{real}}, D' \cup \{x'\})$
        $X^+ \leftarrow S_{\text{swap}}$ samples from $D_{\text{pool\_remaining}}$ with the highest $g_{\text{add}}(x')$
        $D_{\text{curr}} \leftarrow D' \cup \{X^+\}$
    return D_{curr}

We conduct an empirical privacy assessment using the `Synth-MIA` library [12]. This framework executes a broad range of state-of-the-art membership inference attacks (MIAs; see appendix for more details) in a no-box threat scenario, where an attacker has access to the synthetic data but no knowledge of the generation process. As proposed by Ward et al. [12], we compute the maximum MIA score (AUC and TPR at various low FPR) across all implemented attacks, and report the difference between baseline and optimized synthetic dataset as *privacy loss* in Table 1, once for AUC and once for TPR@FPR=0. The true positive rate (TPR) at a low false positive rate (FPR) is used because it reflects an attacker's effectiveness under realistic, privacy-relevant conditions where only a few false positives can be tolerated.

3 Results

A summary of our results is presented in Table 1. The full tables can be found in the appendix (Tables 3, 4, and 5). Our results show a clear improvement in the fidelity of data optimized by ISSOSYNTH for all datasets and generators. The magnitude of this improvement varies by generator. The gains are most substantial for CTGAN and TVAE, with improvements as high as +0.13 (CTGAN on Nursery) and +0.12 (TVAE on Electricity). For other generators, such as TabularARGN, the fidelity gains are more modest but still consistently positive (e.g., +0.04 on Electricity and +0.03 on Adult).

In the downstream utility, there is an overall trend visible that by improving the fidelity, ISSOSYNTH also improves the AUC score across most datasets and SDGs. For the Nursery dataset in particular,

Table 1: Summary of ISSOSYNTH performance for all datasets. The fidelity gain and utility gain show the improvement over the baseline generator's output. Privacy loss columns show the difference in maximum MIA scores between the optimized and baseline data, where values near zero indicate no significant change in empirical privacy risk.

Dataset	Synthetic data generator	Fidelity gain	Utility gain	Privacy loss (AUC)	Privacy loss (TPR@FPR=0)
Adult	CTGAN	0.09	0.01	0.003	0.001
	Gaussian Copula	0.07	0.02	0.001	0.001
	TabularARGN	0.03	0.01	0.016	0.000
	TVAE	0.06	0.01	0.003	-0.001
Bank Marketing	CTGAN	0.10	0.03	0.003	-0.001
	Gaussian Copula	0.06	-0.01	0.000	-0.001
	TabularARGN	0.02	0.02	0.014	0.000
	TVAE	0.07	0.02	0.001	0.000
Electricity	CTGAN	0.10	0.02	-0.001	-0.001
	Gaussian Copula	0.07	0.01	0.000	0.000
	TabularARGN	0.04	0.01	0.004	0.000
	TVAE	0.12	0.02	-0.003	0.000
Nursery	CTGAN	0.13	0.08	0.002	0.001
	Gaussian Copula	0.03	0.26	-0.003	0.000
	TabularARGN	0.01	0.01	-0.003	-0.002
	TVAE	0.05	0.03	-0.002	-0.001

we see a significant improvement, especially for Gaussian Copula and CTGAN generators. We hypothesize this is because Nursery is the only multi-class dataset, presenting a complexity that weaker generators fail to capture. Our fidelity-based selection process seems to successfully identify and preserve the rare, high-utility samples.

The gains in fidelity and utility are achieved without any significant privacy loss, i.e., increase in vulnerability to state-of-the-art membership inference attacks. There are only two instances with a privacy loss of ISSOSYNTH above 1%, but given that the TPR@FPR=0 does not increase, these differences are unlikely to be leveraged effectively in a real-world attack. In general, the privacy evaluation in Table 5 in the appendix shows that all MIA AUC scores remain close to 0.5 for both baseline and optimized data, and the TPR@FPR=0 is close to 0, both indicating that attacks are unable to determine membership much better than random guessing.

4 Related Work

The idea of generating an oversampled pool of synthetic data and then selecting a high-quality subset was also explored in a concurrent work by Hahn et al. [3]. In contrast to our approach, they employ a genetic algorithm. Post-processing synthetic data to enhance utility has also been proposed in the context of differential privacy [11]. However, their approach relies on re-weighting rather than using oversampling and subset selection and focuses on numerical features.

5 Conclusion

We introduced ISSOSYNTH, a generator-agnostic oversampling and iterative subset selection method designed to enhance the fidelity of synthetic tabular data. Our evaluation demonstrates that our approach improves fidelity and utility on downstream tasks without notably increasing privacy risks through membership inference attacks. The method proved particularly effective for datasets with complex distributions and feature combinations. Future work could explore integrating formal differential privacy guarantees into the selection process or adapting the optimization metric to target specific utility or fairness objectives.

Acknowledgments and Disclosure of Funding

The development of the proposed method was carried out independently, while the evaluation phase was partially supported by the project PRO'k'RESS managed by the Austrian Research Promotion Agency FFG (grant number 60155996). Know Center Research GmbH is a COMET competence center that is financed by the Austrian Federal Ministry of Innovation, Mobility and Infrastructure (BMIMI), the Austrian Federal Ministry of Economy, Energy and Tourism (BMWET), the Province of Styria, the Steirische Wirtschaftsförderungsgesellschaft m.b.H. (SFG), the Vienna business agency and the Standortagentur Tirol. The COMET programme is managed by the Austrian Research Promotion Agency FFG.

References

[1] Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

[2] João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. In Ana L. C. Bazzan and Sofiane Labidi, editors, *Advances in Artificial Intelligence – SBIA 2004*, pages 286–295, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[3] Waldemar Hahn, Martin Sedlmayr, and Markus Wolfien. Genetic algorithm for subset selection in synthetic tabular data. In *2025 International Conference on Artificial Intelligence, Computer, Data Sciences and Applications (ACDSA)*, pages 1–6. IEEE, 2025.

[4] Rita P. Moro, S. and P. Cortez. Bank Marketing. UCI Machine Learning Repository, 2014. DOI: https://doi.org/10.24432/C5K306.

[5] Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. The synthetic data vault. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410, Oct 2016.

[6] 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*, 12:2825–2830, 2011.

[7] Vladislav Rajkovic. Nursery. UCI Machine Learning Repository, 1989. DOI: https://doi.org/10.24432/C5P88W.

[8] Andrey Sidorenko, Michael Platzer, Mario Scriminaci, and Paul Tiwald. Benchmarking synthetic tabular data: A multi-dimensional evaluation framework, 2025.

[9] Paul Tiwald, Ivona Krchova, Andrey Sidorenko, Mariana Vargas Vieyra, Mario Scriminaci, and Michael Platzer. Tabularargn: A flexible and efficient auto-regressive framework for generating high-fidelity synthetic data. *arXiv preprint arXiv:2501.12012*, 2025.

[10] Boris Van Breugel, Hao Sun, Zhaozhi Qian, and Mihaela van der Schaar. Membership inference attacks against synthetic data through overfitting detection. *arXiv preprint arXiv:2302.12580*, 2023.

[11] Hao Wang, Shivchander Sudalairaj, John Henning, Kristjan Greenewald, and Akash Srivastava. Post-processing private synthetic data for improving utility on selected measures. *Advances in Neural Information Processing Systems*, 36:64139–64154, 2023.

[12] Joshua Ward, Xiaofeng Lin, Chi-Hua Wang, and Guang Cheng. Synth-mia: A testbed for auditing privacy leakage in tabular data synthesis, 2025.

[13] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32, 2019.

A Experimental Details

A.1 Datasets

We use four publicly available datasets. The **Adult** dataset contains census data to predict income. **Bank Marketing** contains data from a direct marketing campaign. **Electricity** contains electricity market data. **Nursery** contains data from a nursery school admissions model. The Nursery dataset contained one target class that included only two samples, which we removed from our experiments. Table 2 shows the overall number of rows (before splitting into training, test and reference set), features and evaluation target of the datasets.

Table 2: Overview of datasets used in the evaluation.

Name	Number of Rows	Features	Target
Adult	48,842	14	Binary
Bank Marketing	45,211	16	Binary
Electricity	45,312	8	Binary
Nursery	12,958	8	Multi-class (4)

A.2 Implementation Details

Generators For CTGAN, TVAE and Gaussian Copula, we used the default hyperparameters from the `Synthetic Data Vault` [5] library. For TabularARGN, we used the official implementation[2] with a maximal training time of 10 minutes. For all generators, we created an oversampled synthetic pool 10 times the size of the training set.

Computing $\mathcal{L}_1$ Distance To efficiently compute the $\mathcal{L}_1$ distance across distributions, all features are first discretized into 10 bins. For numerical features, a quantile-based binning approach divides the data into 10 equally populated groups. Categorical features are binned on their frequency, mapping the 9 most frequent categories to their own bins and collapsing all other less frequent categories into a single "other" bin. All subsequent comparisons (uni-, bi-, and trivariate) are performed by calculating histograms and their L_1 distances on this binned integer representation.

Efficient Gain Computation A naive implementation of the gain calculations in Algorithm 1 (for `GreedyTrimming` and `Refinement`) would be computationally prohibitive, as it implies re-calculating the full $\mathcal{L}_1$ metric for every candidate sample.

We pre-compute all target contingency tables from D_{real} and maintain running contingency tables for the current subset D_{curr}. To calculate the removal gain $g(x)$ for a sample x, we do not re-compute $\mathcal{L}_1(D_{\text{real}}, D_{\text{curr}} \setminus \{x\})$. Instead, for each monitored marginal, we identify the sample's bin b and calculate the change to the L_1 error for that single bin. Removing a sample improves the score (a positive gain) if its bin is overfull, but worsens the score (a penalty) if its bin is underfull.

This entire process is vectorized, allowing us to compute gains for all batch candidates simultaneously. A similar vectorized approach is used for computing the addition gain. This enables an efficient execution of ISSOSYNTH even for large datasets. For example, processing the generated synthetic pool for the adult dataset with over 160,000 samples takes approximately 20 seconds on a machine with an Intel Core i7-10875H processor and 32 GB of RAM to run.

ISSOSYNTH The full implementation of ISSOSYNTH is available at `https://github.com/Gandagorn/mostlyai_flat`. For our evaluation, ISSOSYNTH was configured with the following hyperparameters:

- **IPF** The initial subset was selected to be $1.2\times$ the target size ($N_{\text{sample}} = 1.2 \times N$). We used the top $k_{\text{pairs}} = 100$ bivariate pairs for fitting and $K_1 = 6$ iterations.
- **Trimming** The swap size S_{trim} was adaptive: $\min(100, N/100)$. The optimization targets were all univariate, top 50 bivariate and top 5 trivariate distributions.

[2] `https://github.com/mostly-ai/mostlyai-engine`

Table 3: Full fidelity evaluation. Statistical Fidelity is the mean of uni-, bi-, and trivariate accuracy scores, comparing synthetic data to the real training set. 'Synthetic Data' refers to the data directly sampled from the generator, and 'Optimized Data' refers to the dataset selected by ISSOSYNTH. Results are (mean $\pm$ std) over 3 random splits.

		Accuracy	
Dataset	Synthetic data generator	Synthetic Data	Optimized Data
Adult	CTGAN	0.84 ± 0.00	0.93 ± 0.01
	Gaussian Copula	0.66 ± 0.01	0.73 ± 0.01
	TabularARGN	0.95 ± 0.01	0.98 ± 0.00
	TVAE	0.79 ± 0.01	0.86 ± 0.02
Bank Marketing	CTGAN	0.84 ± 0.01	0.94 ± 0.00
	Gaussian Copula	0.70 ± 0.01	0.76 ± 0.01
	TabularARGN	0.96 ± 0.00	0.98 ± 0.00
	TVAE	0.79 ± 0.02	0.86 ± 0.02
Electricity	CTGAN	0.73 ± 0.00	0.82 ± 0.01
	Gaussian Copula	0.59 ± 0.06	0.67 ± 0.06
	TabularARGN	0.93 ± 0.00	0.97 ± 0.00
	TVAE	0.79 ± 0.02	0.91 ± 0.01
Nursery	CTGAN	0.83 ± 0.01	0.97 ± 0.01
	Gaussian Copula	0.95 ± 0.00	0.98 ± 0.00
	TabularARGN	0.97 ± 0.00	0.98 ± 0.00
	TVAE	0.89 ± 0.04	0.93 ± 0.04

- **Refinement** We ran for $K_2 = 50$ iterations. The swap size S_{swap} was adaptive (same as trimming). The optimization targets were all univariate, top 50 bivariate, and top 5 trivariate distributions.

Downstream Models For all downstream prediction experiments we use the gradient boosting classifier model of the `scikit-learn` library [6].

Membership Inference Attacks Membership inference attacks aim to determine whether specific records were part of a model's training data. They are used for empirical privacy evaluation because their success indicates potential leakage of sensitive information. We use the `Synth-MIA` testbed [12] as proposed by the authors by running the implemented membership inference attacks and reporting the maximum MIA AUC and TPR at low FPR. Therefore, we included the following attacks: Distance to Closest Record (DCR) attacks, Generative Model Likelihood Ratio Attack (GenLRA), Data Plagiarism Index (DPI), Loss-based GAN (LOGAN) attack, DOMIAS attack [10], Monte Carlo-based (MC) attack, Density Estimation attack, Local Neighborhood attack, and classifier attack. Some of these attacks assume that the attacker has access to a reference dataset, which is distinct from the targeted training dataset but comes from the same distribution. We used one of the three equally sized dataset splits as the reference dataset, while the other two were used as training and test set.

B Full tables

Tables 3, 4 and 5 show the full results of the fidelity, utility and privacy evaluation respectively.

Table 4: Full downstream utility evaluation. We report the AUC-ROC (mean $\pm$ std) of a gradient boosting model. The model was trained on synthetic data (baseline or optimized) and evaluated on the test set. 'Synthetic Data' refers to the data directly sampled from the generator, and 'Optimized Data' refers to the dataset selected by ISSOSYNTH.

		AUC	
Dataset	Synthetic Data Generator	Synthetic Data	Optimized Data
Adult	CTGAN	0.88 ± 0.01	0.89 ± 0.01
	Gaussian Copula	0.81 ± 0.03	0.83 ± 0.02
	TabularARGN	0.91 ± 0.00	0.91 ± 0.00
	TVAE	0.88 ± 0.01	0.88 ± 0.01
Bank Marketing	CTGAN	0.82 ± 0.03	0.85 ± 0.01
	Gaussian Copula	0.81 ± 0.02	0.80 ± 0.02
	TabularARGN	0.89 ± 0.00	0.90 ± 0.00
	TVAE	0.83 ± 0.02	0.85 ± 0.01
Electricity	CTGAN	0.79 ± 0.01	0.81 ± 0.01
	Gaussian Copula	0.78 ± 0.06	0.78 ± 0.06
	TabularARGN	0.87 ± 0.00	0.88 ± 0.00
	TVAE	0.83 ± 0.02	0.85 ± 0.02
Nursery	CTGAN	0.90 ± 0.04	0.98 ± 0.00
	Gaussian Copula	0.68 ± 0.08	0.95 ± 0.03
	TabularARGN	0.98 ± 0.00	0.99 ± 0.00
	TVAE	0.93 ± 0.06	0.96 ± 0.03

Table 5: Full privacy evaluation. We report the maximum MIA scores (mean ± std over 3 runs) across all attackers. 'Synthetic' refers to the dataset directly sampled from the generator, and 'Optimized' refers to the dataset selected by ISSOSYNTH.

		AUC		TPR@FPR=0		TPR@FPR=0.1	
Dataset	Generator	Synthetic	Optimized	Synthetic	Optimized	Synthetic	Optimized
Adult	CTGAN	0.508 ± 0.003	0.510 ± 0.002	0.000 ± 0.000	0.001 ± 0.000	0.110 ± 0.004	0.109 ± 0.004
	Gaussian Copula	0.506 ± 0.002	0.507 ± 0.002	0.000 ± 0.000	0.000 ± 0.000	0.106 ± 0.003	0.106 ± 0.003
	TabularARGN	0.520 ± 0.008	0.536 ± 0.002	0.000 ± 0.000	0.001 ± 0.000	0.120 ± 0.008	0.124 ± 0.003
	TVAE	0.503 ± 0.001	0.505 ± 0.002	0.000 ± 0.000	0.000 ± 0.000	0.105 ± 0.002	0.108 ± 0.001
Bank Marketing	CTGAN	0.508 ± 0.004	0.511 ± 0.004	0.000 ± 0.000	0.000 ± 0.000	0.109 ± 0.003	0.112 ± 0.003
	Gaussian Copula	0.505 ± 0.002	0.505 ± 0.004	0.000 ± 0.000	0.000 ± 0.000	0.108 ± 0.001	0.109 ± 0.005
	TabularARGN	0.520 ± 0.003	0.534 ± 0.005	0.000 ± 0.000	0.000 ± 0.000	0.114 ± 0.004	0.125 ± 0.003
	TVAE	0.507 ± 0.003	0.508 ± 0.001	0.001 ± 0.000	0.000 ± 0.000	0.108 ± 0.001	0.111 ± 0.002
Electricity	CTGAN	0.508 ± 0.002	0.507 ± 0.004	0.001 ± 0.000	0.001 ± 0.000	0.106 ± 0.001	0.109 ± 0.006
	Gaussian Copula	0.506 ± 0.002	0.505 ± 0.003	0.001 ± 0.000	0.001 ± 0.000	0.107 ± 0.002	0.110 ± 0.004
	TabularARGN	0.510 ± 0.003	0.513 ± 0.005	0.001 ± 0.000	0.001 ± 0.000	0.108 ± 0.002	0.112 ± 0.005
	TVAE	0.515 ± 0.002	0.512 ± 0.002	0.001 ± 0.000	0.000 ± 0.000	0.113 ± 0.006	0.111 ± 0.002
Nursery	CTGAN	0.517 ± 0.009	0.519 ± 0.009	0.002 ± 0.002	0.003 ± 0.001	0.124 ± 0.005	0.126 ± 0.007
	Gaussian Copula	0.516 ± 0.009	0.513 ± 0.008	0.003 ± 0.002	0.003 ± 0.002	0.117 ± 0.001	0.106 ± 0.005
	TabularARGN	0.515 ± 0.005	0.512 ± 0.013	0.003 ± 0.001	0.002 ± 0.000	0.124 ± 0.015	0.112 ± 0.008
	TVAE	0.518 ± 0.014	0.516 ± 0.008	0.003 ± 0.002	0.002 ± 0.000	0.123 ± 0.017	0.129 ± 0.004