BENCHMARKING DIFFERENTIALLY PRIVATE TABULAR DATA SYNTHESIS ALGORITHMS

Kai Chen, Xiaochen Li & Chen Gong
Department of Computer Science
University of Virginia
Charlottesville, VA 22903, USA
{kaichen,xiaochenli,fzv6en}@virginia.edu

Ryan Mckenna
Google Research
Seattle, WA 98103, USA
mckennar@google.com

Tianhao Wang
Department of Computer Science
University of Virginia
Charlottesville, VA 22903, USA
tianhao@virginia.edu

ABSTRACT

Differentially private (DP) tabular data synthesis algorithms generate artificial data that preserves the statistical properties of private data while safeguarding individual privacy. However, the emergence of diverse algorithms in recent years has introduced challenges in practical applications, such as inconsistent data processing methods, and the lack of in-depth algorithm comparisons and analysis. These factors create significant obstacles to selecting appropriate algorithms. In this paper, we address these challenges by proposing a novel benchmark for evaluating tabular data synthesis methods. We present a unified evaluation framework that integrates data preprocessing, feature selection, and data synthesis modules, facilitating fair and comprehensive comparisons. Our evaluation reveals that no single method consistently outperforms the rest across all scenarios. Furthermore, we conduct an in-depth experimental evaluation of each algorithmic module, offering insights into the strengths and limitations of different strategies. This lays the foundation for designing more robust and interpretable methods for private data synthesis. Source codes are available at the link[1].

1 INTRODUCTION

Private tabular data synthesis algorithms generate artificial data that preserves the statistical properties of real data while protecting individual privacy. Differential privacy (DP) is one of the gold standards for protecting privacy. DP ensures that the inclusion or exclusion of any single data point does not significantly affect the outcome, thereby protecting each individual data point within a dataset. A substantial body of research has been proposed to address the tabular data synthesis problem with DP. These can be broadly classified into two categories: statistical methods and machine learning methods. Statistical methods (Zhang et al., 2017; Vietri et al., 2022; Zhang et al., 2021; Liu et al., 2023; Zhang et al., 2017; McKenna et al., 2021; Vietri et al., 2020; Hardt et al., 2012) compress data information through statistical properties, such as low-dimensional data distributions, to achieve data generation. On the other hand, machine learning methods leverage deep learning frameworks designed for data generation, such as generative networks (Liu et al., 2021; Harder et al., 2021; Jordon et al., 2018; Xie et al., 2018; Torkzadehmahani et al., 2019) and diffusion models (Kotelnikov et al., 2023; Pang et al., 2024; Li et al., 2024).

While several studies have been conducted on tabular data synthesis, we still face several challenges: (1) *Lack of unified evaluation settings.* Beyond algorithmic strategies, evaluation settings, such as data preprocessing, play a significant role in determining algorithm performance. However, these settings are often considered trivial and thus are frequently overlooked in many methods,

[1] https://github.com/KaiChen9909/tab_bench

which potentially leads to unfair comparisons between methods. (2) *Lack of comprehensive and in-depth evaluation and analysis.* Due to various reasons, such as concurrent development or relatively recent introduction, comparisons between recently proposed methods and existing works remain incomplete, particularly between some representative methods like RAP++ (Vietri et al., 2022), Private-GSD (Liu et al., 2023) and AIM (McKenna et al., 2022). Moreover, current works often focus on proposing new methods, only providing limited intuition and evaluation of algorithms. Consequently, certain deeper aspects, such as the utility of individual algorithm modules, remain underexplored. For example, questions such as how to select features to represent the dataset more accurately or what limitations different synthesis strategies face given selected features are insufficiently addressed.

In light of the above challenges, we believe proposing a new benchmark for evaluating tabular data synthesis is necessary. The contributions of our benchmark work are as follows.

Proposing a Unified Framework for Evaluation. We first propose a generalized framework and align all methods within this framework to ensure fairness and objectivity in comparisons. The framework consists of a *data preprocessing module*, a *feature selection module*, and a *data synthesis module*. Notably, this is the first framework to explicitly consider the impact of preprocessing on algorithm comparisons. Additionally, dividing the workflow of current algorithms into feature selection and data generation modules provides a modular perspective that enhances our understanding and evaluation of these methods.

Providing a Comprehensive Comparison and Analysis. In our experiments, we include current state-of-the-art methods (McKenna et al., 2022; Zhang et al., 2021; Cai et al., 2021; Vietri et al., 2022; Liu et al., 2021; Harder et al., 2021) under both statistical and machine learning methods, along with newly proposed methods (Liu et al., 2023; Kotelnikov et al., 2023) that have not been thoroughly explored in prior research. Moreover, based on the unified framework, our experiments include the evaluation of different algorithmic modules, which are more fine-grained and help us analyze how they function independently.

We observed some important experimental findings, i.e., (1) no single method dominates across all experimental settings. Generally, statistical methods outperform machine learning methods in terms of accuracy. However, machine learning methods are significantly more time-efficient. (2) Preprocessing plays a crucial role in enhancing the efficiency of algorithms without introducing substantial synthesis errors, supporting our attention on this step. (3) For the selection module, adaptive selection algorithms with well-designed measurement techniques outperform others. For the data synthesis module, all current algorithms exhibit notable limitations in different ways.

2 DP Tabular Data Synthesis

2.1 Problem Formulation

Differential privacy (DP) has become the *de facto* standard for data privacy. It allows aggregated statistical information to be extracted while limiting the disclosure of information about individuals. More formally, the definition of DP is given by

Definition 1 (Differential Privacy) *An algorithm $\mathcal{A}$ satisfies (ε, δ)-differential privacy ((ε, δ)-DP) if and only if for any two neighboring datasets D and D' and any $T \subseteq$ Range$(\mathcal{A})$, we have*

$$\texttt{Pr}\left[\mathcal{A}(D) \in T\right] \leq e^{\varepsilon}\, \texttt{Pr}\left[\mathcal{A}(D') \in T\right] + \delta.$$

We say two datasets are neighboring ($D \simeq D'$) when they differ on one tuple/sample, i.e., one dataset can be obtained from the other by removing one tuple.

In this work, we focus our attention on DP mechanisms for tabular data synthesis, which are designed to generate an artificial tabular dataset that mirrors the statistical characteristics of the original dataset without compromising individual privacy. More formally, assuming that we have a tabular dataset D composed of n records $\{x_1, \cdots, x_n\}$, and each record has d attributes $\{A_1, \cdots, A_d\}$, we want to generate a dataset D_s similar to D. Here, the synthetic dataset D_s is said to be similar to D if $f(D_s)$ is close to $f(D)$ for a broad class of functions f.

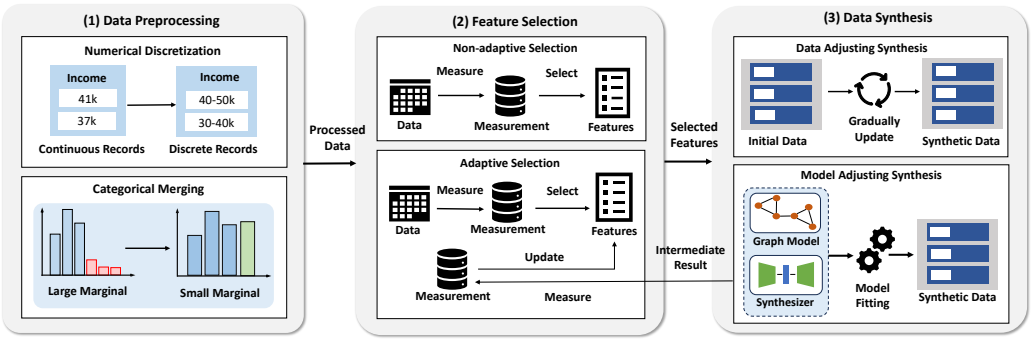


Figure 1: The proposed unified framework. The three blocks represent data preprocessing, feature selection, and data synthesis steps respectively.

2.2 Existing Work

We start by briefly reviewing current state-of-the-art algorithms (a detailed summary of existing work is provided in Appendix A.1). PrivSyn (Zhang et al., 2021) first select a set of low-dimensional marginals and then synthesize data by updating an initialized dataset to align with those marginals. (Cai et al., 2021) introduced PrivMRF, and (McKenna et al., 2022) proposed AIM, both of which dynamically select low-dimensional marginals and employ PGMs for synthesis. RAP++ (Vietri et al., 2022) and Private-GSD (Liu et al., 2023) also apply an adaptive marginal selection mechanism, but their synthesis algorithms are constructed on relaxed projection and genetic algorithms.

Besides statistical methods, machine learning methods also show diverse working principles. GEM (Liu et al., 2021) combines deep generative networks with adaptive marginal selection mechanisms, while DP-MERF (Harder et al., 2021) employs random Fourier feature loss to train generative networks. Additionally, TabDDPM (Kotelnikov et al., 2023) leverages diffusion models' representational power to fit target data directly. While TabDDPM was not originally designed for DP, it achieves state-of-the-art performance among non-DP methods and can be adapted for DP synthesis using DP-SGD (Du & Li, 2024). Consequently, we include TabDDPM in our analysis.

2.3 Unified Framework

Current works often focus on end-to-end comparisons, which cannot easily attribute performance differences to certain parts of the algorithms. Therefore, similar to some previous works (Liu et al., 2021; McKenna et al., 2021; Hu et al., 2023), we propose our algorithm framework, as shown in Figure 1. This framework not only serves as a way to understand existing algorithms, but also allows us to combine modules from different prior works, leading to new mechanisms and enabling the evaluation of individual algorithmic modules. It consists of three modules:

Preprocessing Module. As observed in our evaluation (see Table 2), many datasets contain attributes with large domain sizes (e.g., exceeding 10^5). These high-cardinality attributes lead to large marginals or huge model dimensions, posing significant challenges for data synthesis. However, these challenges can be alleviated by proper preprocessing steps, compressing the domains of such attributes and thus facilitating faster synthesis. However, preprocessing is often ignored for simplicity, and several prior works discussed have not specified this aspect explicitly, which may impact the fairness of comparisons across methods.

Feature Selection Module. While some algorithms, such as TabDDPM, can fit the entire dataset by their powerful models. It is still infeasible for most methods to estimate the full joint distribution because the domain of the whole dataset increases exponentially with the number of attributes. A more common and practical approach is to utilize some representative local data features to approximate the full joint distribution. For instance, GEM and most statistical methods utilize low-dimensional marginals, and DP-MERF uses the random Fourier feature. Therefore, the second step in the framework is to select representative features. Many different methods have been proposed for this based on similar principles. Generally, we divide them into two categories: *non-adaptive feature selection*,

Table 1: Overall results of synthetic data under different methods. The results with the best performance are highlighted in bold. Ground truth is obtained by comparing real data with test data, which serves as the reference for evaluation.

Dataset	**ACSincome**			**ACSemploy**			**Bank**			**Higgs-small**			**Loan**		
ML Efficiency	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$
PrivSyn	0.38	0.39	0.41	0.42	0.43	0.39	0.47	0.47	0.47	0.40	0.43	0.43	0.25	0.26	0.26
PrivMRF	0.73	0.78	0.78	0.72	0.80	0.81	0.62	0.69	**0.71**	0.50	0.64	0.64	**0.52**	**0.52**	**0.52**
RAP++	0.66	0.73	0.77	0.74	0.77	0.80	0.65	0.69	0.67	0.52	0.53	0.54	0.45	0.42	0.43
AIM	**0.76**	**0.78**	**0.78**	**0.78**	**0.80**	**0.81**	**0.67**	**0.71**	0.71	**0.63**	**0.64**	**0.67**	0.52	0.52	0.52
Private-GSD	0.76	0.77	0.77	0.72	0.73	0.72	0.47	0.48	0.49	0.48	0.47	0.49	0.25	0.24	0.25
GEM	0.70	0.68	0.66	0.67	0.70	0.69	0.51	0.56	0.53	0.51	0.52	0.52	0.50	0.49	0.51
DP-MERF	0.65	0.67	0.71	0.58	0.70	0.66	0.60	0.57	0.55	0.53	0.56	0.57	0.32	0.18	0.18
TabDDPM	0.41	0.41	0.39	0.48	0.42	0.51	0.47	0.47	0.47	0.36	0.35	0.34	0.24	0.24	0.24
Ground Truth	0.79	0.79	0.79	0.81	0.81	0.81	0.76	0.76	0.76	0.72	0.72	0.72	0.54	0.54	0.54
Query Error	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$
PrivSyn	0.003	0.002	0.002	0.006	0.004	0.004	0.007	0.004	0.003	0.009	0.004	0.003	0.006	0.005	0.004
PrivMRF	0.002	0.001	0.001	0.004	0.002	0.002	**0.005**	0.003	0.003	0.005	0.005	0.003	**0.005**	**0.005**	**0.004**
RAP++	0.019	0.005	0.003	0.029	0.009	0.003	0.014	0.006	0.005	0.035	0.029	0.028	0.020	0.014	0.011
AIM	**0.002**	**0.001**	**0.001**	**0.004**	**0.002**	**0.001**	0.007	**0.002**	**0.002**	**0.005**	**0.003**	**0.003**	0.005	0.005	0.004
Private-GSD	0.004	0.003	0.002	0.026	0.026	0.026	0.044	0.044	0.043	0.044	0.044	0.044	0.038	0.037	0.036
GEM	0.014	0.017	0.016	0.010	0.006	0.006	0.118	0.021	0.022	0.066	0.065	0.065	0.030	0.030	0.029
DP-MERF	0.019	0.018	0.024	0.039	0.037	0.036	0.038	0.035	0.036	0.039	0.035	0.034	0.006	0.006	0.006
TabDDPM	0.066	0.064	0.060	0.088	0.067	0.079	0.074	0.071	0.088	0.106	0.106	0.107	0.067	0.066	0.070
Ground Truth	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
Fidelity Error	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$	$\varepsilon=0.2$	$\varepsilon=1$	$\varepsilon=5$
PrivSyn	0.15	0.12	0.12	0.12	0.09	0.09	0.24	0.10	0.21	0.29	0.20	0.15	0.34	0.35	0.36
PrivMRF	0.11	0.07	0.05	0.07	0.04	0.03	0.13	**0.07**	**0.06**	0.21	**0.16**	0.16	**0.31**	**0.24**	**0.23**
RAP++	0.52	0.24	0.19	0.30	0.13	0.07	0.43	0.36	0.36	0.69	0.65	0.65	0.66	0.58	0.55
AIM	**0.09**	**0.06**	**0.05**	**0.05**	**0.03**	**0.02**	**0.11**	0.09	0.09	**0.19**	0.17	**0.14**	0.35	0.32	0.29
Private-GSD	0.23	0.21	0.20	0.22	0.22	0.22	0.52	0.52	0.52	0.65	0.65	0.65	0.67	0.66	0.66
GEM	0.26	0.28	0.27	0.15	0.08	0.09	0.76	0.21	0.23	0.57	0.57	0.36	0.53	0.52	0.52
DP-MERF	0.52	0.51	0.50	0.34	0.34	0.32	0.47	0.48	0.48	0.60	0.56	0.54	0.91	0.92	0.93
TabDDPM	0.78	0.77	0.71	0.60	0.51	0.57	0.79	0.78	0.82	0.70	0.81	0.88	0.95	0.95	0.94
Ground Truth	0.05	0.05	0.05	0.02	0.02	0.02	0.03	0.03	0.03	0.12	0.12	0.12	0.10	0.10	0.10

which finishes selection in one step like PrivSyn, and *adaptive feature selection*, which incorporates iterative refinement for the selection result such as AIM, PrivMRF, RAP++ and GEM.

Data Synthesis Module. The third step in the framework is to synthesize data that aligns well with the private feature measurements taken in the previous step. Broadly, we categorize existing synthesis methods into two types based on their synthesis strategies: *data-adjusting methods*, which gradually update records in an initialized dataset and finally output this dataset as the synthetic result, and *model-adjusting methods*, which fit generative models and synthesize data by these models. It is also notable that these two categories are not mutually exclusive. We can regard an initialized dataset as a model where each data record is a model parameter, so adjusting data can also be viewed as adjusting a model. The distinction here is more for demonstrating algorithm intuition.

3 EXPERIMENTS

We consider eight state-of-the-art methods: PrivSyn (Zhang et al., 2021), PrivMRF (Cai et al., 2021), AIM (McKenna et al., 2022), RAP++ (Vietri et al., 2022), Private-GSD (Liu et al., 2023), GEM (Liu et al., 2021), DP-MERF (Harder et al., 2021) and TabDDPM (Kotelnikov et al., 2023). We choose five datasets used in previous work (Liu et al., 2023; McKenna et al., 2022; Zhang et al., 2021; Kotelnikov et al., 2023) as our datasets, which are ACSincome (INC), ACSemploy (EMP), Bank (BK), Higgs-small (HIG) and Loan (LN). By default, we use uniform binning and category merging as our preprocessing method. Detailed information on datasets, evaluation metrics, and algorithm implementation is provided in the appendix.

3.1 OVERALL EVALUATION

Table 1 shows the detailed machine learning efficiency, query error, and fidelity error of different methods, respectively. To make the result more straightforward, we utilize the t-SNE (Van der Maaten & Hinton, 2008), a dimensionality reduction technique, to visualize the synthesis results of all methods on the Bank dataset in Figure 5.

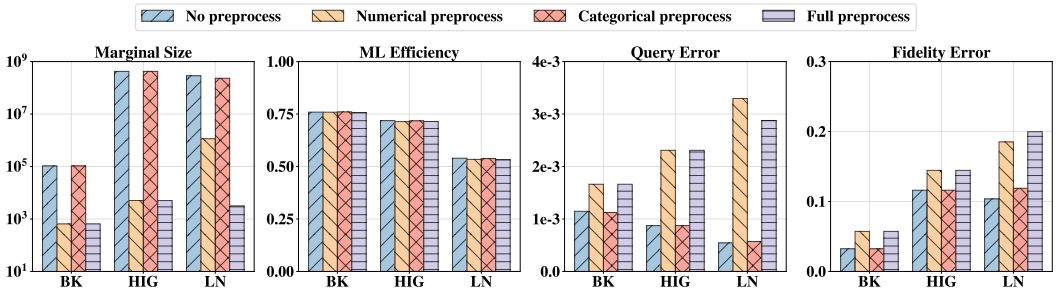


Figure 2: Metrics under different preprocessing settings. These metrics are obtained by comparing preprocessed raw data with test data. Numerical preprocess and categorical preprocess mean only conducting preprocessing on numerical and categorical attributes respectively. By default, the results are under the setting that $\varepsilon = 1.0$ and 10% of the budget is used for preprocessing (if needed).

No method dominates. The first clear conclusion is that no method completely dominates. In Table 1, PrivMRF and AIM show an outstanding performance in machine learning efficiency. Moreover, these two methods with PrivSyn also perform well on query error and fidelity error. In Figure 5, the synthetic results by PrivSyn, PrivMRF, RAP++, AIM, and Private-GSD all demonstrate relatively similar distributions to the real data.

Statistical methods outperform machine learning methods in utility but are more time-consuming. Another straightforward finding is that statistical methods generally perform better than machine learning approaches. The best overall performance in metrics belongs to two statistical methods, AIM and PrivMRF, while the quantitative evaluations of GEM, DP-MERF, and TabDDPM are worse than most statistical methods. Moreover, the t-SNE plot results of some machine learning methods are also unsatisfactory. For instance, in Figure 5, DP-MERF and TabDDPM's synthetic data do not match the real data distribution well. The advantage of machine learning methods is also noteworthy. We present the running times of these algorithms in Table 15. On average, most machine learning-based methods are time-efficient (partially because they use GPU in model-fitting steps).

3.2 Preprocessing Investigation

Preprocessing can reduce data complexity with small errors. Directly comparing the algorithms' performances with and without preprocessing is difficult because running algorithms on raw datasets is often too time-consuming and computationally complex (some methods even require more than 24 hours to run on raw datasets). Therefore we consider comparing the marginal sizes and utility metrics of preprocessed and raw datasets. The detailed results are shown in Figure 2.

A straightforward finding is that preprocessing decreases the complexity of data, only with the introduction of a small error. In Figure 2, the average marginal size significantly decreases after preprocessing (from 10^8 to 10^3 in Higgs-small and Loan datasets). Meanwhile, its negative influences on utility are small enough (change on query error < 0.003, TVD < 0.1). We infer this as binning can preserve most numerical characteristics, and the low-frequency categorical values only contribute a small proportion of the overall correlation between attributes in the dataset.

3.3 Module Comparison

In this subsection, we respectively evaluate different marginal selection modules and synthesis modules. In other words, we fix either the selection or the synthesis approach and then evaluate how different algorithms perform under that fixed condition. We divide the datasets into two groups. The first group, called "small datasets", ACSincome and ACSemploy, is relatively low-dimensional with moderately sized attributes' domains. The second group, called "large datasets", including Bank, Higgs-small, and Loan, is higher-dimensional or has larger attribute domains. The results are shown in Figure 3 and Figure 4.

Adaptive strategy with well-designed criterion makes a good selection. We compare different selection methods by combining them with the same synthesis module (in our experiment we lever-

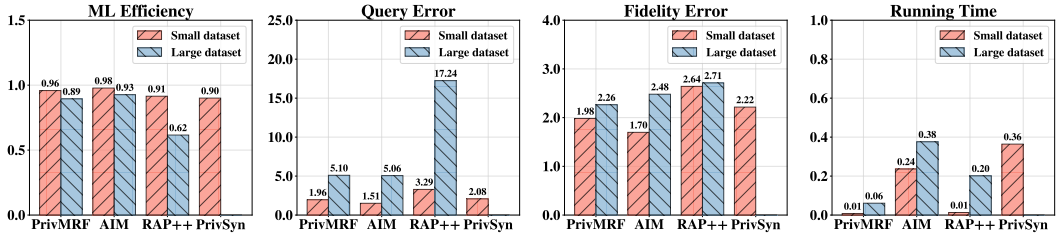


Figure 3: Average scaled metrics for different selection modules on different datasets. We use PGM as the common synthesizer. The machine learning efficiency, query error, and fidelity error are scaled using ground truth values provided in Table 1 while running time is scaled using a simple min-max linear normalization.

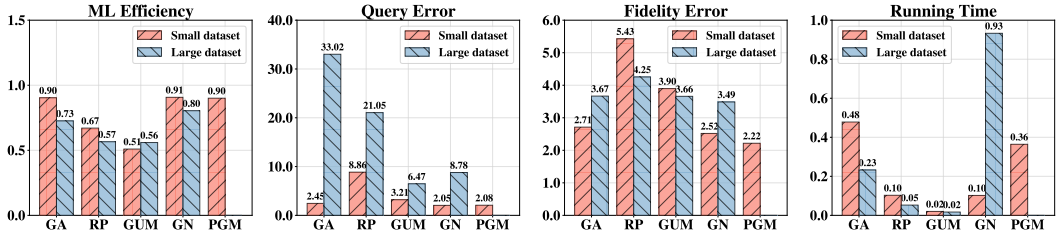


Figure 4: Average scaled metrics for different synthesis modules on different datasets. We use PrivSyn as the common marginal selector.

age PGM). Among the selection methods, PrivSyn and RAP++ show weaker fitting utility. PrivSyn fails to use intermediate information during selection, reducing its ability to capture necessary data features and leading to excessive selections. This also overwhelms PGM on larger datasets due to high memory demands. Similarly, RAP++ employs the Gumbel mechanism to select multiple marginals per iteration without fully accounting for intermediate results. Moreover, its selection criterion focuses solely on marginal query error, ignoring noise error, which is unbalanced and potentially influences the algorithm's performance.

The utility performances of AIM and PrivMRF, are very similar since both perform an adaptive marginal refinement after updating the intermediate information. A notable observation is that PrivMRF has a shorter running time than AIM. We attribute this to the well-designed initial marginal set in PrivMRF, which decreases the need for further refinement and speeds up the total algorithm.

No synthesis module completely outperforms. We compare five synthesis algorithms: GUM, PGM, relaxed projection (RP), genetic algorithm (GA), and generative network (GN), employed by PrivSyn, AIM, RAP++, Private-GSD, and GEM, respectively. To control the running time, we use the PrivSyn selection algorithm as the common marginal selector. The results are shown in Figure 4.

Among these methods, PGM and the generative network achieve the best overall accuracy and stability in fitting features on small datasets. We believe it is because PGM benefits from its expressive structure, which infers marginals without refitting existing ones, thus minimizing compounding errors. However, the PGM's fitting process is quite slow, due to the densely selected marginals by PrivSyn, causing it to fail to deal with large datasets. The generative network's strong representational capacity also supports accurate feature fitting on small datasets. However, when it turns to larger and more complex datasets, its superiority in efficiency and utility will be diminished due to the greater difficulty of training high-dimensional models.

In contrast, some data-adjusting methods, GUM, the genetic algorithm, and relaxed projection, demonstrate weaker fitting capabilities. GUM refines marginals individually, overlooking global correlations. However, it shows a superiority in running time. The genetic algorithm's reliance on randomness makes it unsuitable for complex, high-dimensional datasets with large attribute domains under time constraints, leading to poor performance on large datasets. Relaxed projection suffers similarly; in high-dimensional spaces with extensive attributes' domains, the encoded data dimension becomes excessively large, complicating optimization, and resulting in poor performance.

4 CONCLUSION

Data synthesis under DP remains a critical and active area of research. Despite a growing number of methods being proposed, the field still lacks a fair and comprehensive evaluation. In this paper, we conduct comprehensive comparisons of several methods on a unified framework. Our overall comparisons suggest that no single method is universally superior, while some experiments highlight the critical role of preprocessing. This paper also highlights the essential principles for designing feature selection algorithms and identifies the limitations of current synthesis modules, providing insights for improving DP tabular synthesis.

REFERENCES

Higgs particle dataset, 2016. URL https://www.openml.org/search?type=data&status=active&id=4532.

Loan dataset, 2020. URL https://www.kaggle.com/datasets/devanshi23/loan-data-2007-2014.

Sergul Aydore, William Brown, Michael Kearns, Krishnaram Kenthapadi, Luca Melis, Aaron Roth, and Ankit A Siva. Differentially private query release through adaptive projection. In *International Conference on Machine Learning*, pp. 457–467. PMLR, 2021.

Vincent Bindschaedler, Reza Shokri, and Carl A. Gunter. Plausible deniability for privacy-preserving data synthesis, 2017. URL https://arxiv.org/abs/1708.07975.

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280*, 2022.

Kuntai Cai, Xiaoyu Lei, Jianxin Wei, and Xiaokui Xiao. Data synthesis via differentially private markov random fields. *Proceedings of the VLDB Endowment*, 14(11):2190–2202, 2021.

Chris Chatfield. *Introduction to multivariate analysis*. Routledge, 2018.

Travis Dick, Cynthia Dwork, Michael Kearns, Terrance Liu, Aaron Roth, Giuseppe Vietri, and Zhiwei Steven Wu. Confidence-ranked reconstruction of census microdata from published statistics. *Proceedings of the National Academy of Sciences*, 120(8), February 2023. ISSN 1091-6490. doi: 10.1073/pnas.2218605120. URL http://dx.doi.org/10.1073/pnas.2218605120.

Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Yuntao Du and Ninghui Li. Towards principled assessment of tabular data synthesis algorithms. *arXiv preprint arXiv:2402.06806*, 2024.

Liyue Fan. A survey of differentially private generative adversarial networks. In *The AAAI Workshop on Privacy-Preserving Artificial Intelligence*, volume 8, 2020.

Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Zhiwei Steven Wu. Dual query: Practical private query release for high dimensional data, 2015. URL https://arxiv.org/abs/1402.1526.

Frederik Harder, Kamil Adamczewski, and Mijung Park. Dp-merf: Differentially private mean embeddings with randomfeatures for practical privacy-preserving data generation. In *International conference on artificial intelligence and statistics*, pp. 1819–1827. PMLR, 2021.

Moritz Hardt, Katrina Ligett, and Frank McSherry. A simple and practical algorithm for differentially private data release. *Advances in neural information processing systems*, 25, 2012.

Yuzheng Hu, Fan Wu, Qinbin Li, Yunhui Long, Gonzalo Munilla Garrido, Chang Ge, Bolin Ding, David Forsyth, Bo Li, and Dawn Song. Sok: Privacy-preserving data synthesis, 2023. URL https://arxiv.org/abs/2307.02106.

James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. Pate-gan: Generating synthetic data with differential privacy guarantees. In *International conference on learning representations*, 2018.

Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. Tabddpm: Modelling tabular data with diffusion models. In *International Conference on Machine Learning*, pp. 17564–17579. PMLR, 2023.

Haoran Li, Li Xiong, and Xiaoqian Jiang. Differentially private synthesization of multi-dimensional data using copula functions. In *Advances in database technology: proceedings. International conference on extending database technology*, volume 2014, pp. 475. NIH Public Access, 2014.

Kecen Li, Chen Gong, Zhixiang Li, Yuzhong Zhao, Xinwen Hou, and Tianhao Wang. {PrivImage}: Differentially private synthetic image generation using diffusion models with {Semantic-Aware} pretraining. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 4837–4854, 2024.

Terrance Liu, Giuseppe Vietri, and Steven Z Wu. Iterative methods for private synthetic data: Unifying framework and new methods. *Advances in Neural Information Processing Systems*, 34: 690–702, 2021.

Terrance Liu, Jingwu Tang, Giuseppe Vietri, and Steven Wu. Generating private synthetic data with genetic algorithms. In *International Conference on Machine Learning*, pp. 22009–22027. PMLR, 2023.

Ryan McKenna, Daniel Sheldon, and Gerome Miklau. Graphical-model based estimation and inference for differential privacy. In *International Conference on Machine Learning*, pp. 4435–4444. PMLR, 2019.

Ryan McKenna, Gerome Miklau, and Daniel Sheldon. Winning the nist contest: A scalable and general approach to differentially private synthetic data. *arXiv preprint arXiv:2108.04978*, 2021.

Ryan McKenna, Brett Mullins, Daniel Sheldon, and Gerome Miklau. Aim: An adaptive and iterative mechanism for differentially private synthetic data. *arXiv preprint arXiv:2201.12677*, 2022.

Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, pp. 263–275. IEEE, 2017.

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

Wei Pang, Masoumeh Shafieinejad, Lucy Liu, and Xi He. Clavaddpm: Multi-relational data synthesis with cluster-guided diffusion models. *arXiv preprint arXiv:2405.17724*, 2024.

Mani Srivastava and Moustafa Alzantot. Differentially private dataset release using wasserstein gans. `https://github.com/nesl/nist_differential_privacy_synthetic_data_challenge`, 2019. Accessed 2023-xx-xx.

Arun Sai Suggala and Praneeth Netrapalli. Online non-convex learning: Following the perturbed leader is optimal. In *Algorithmic Learning Theory*, pp. 845–861. PMLR, 2020.

Vasilis Syrgkanis, Akshay Krishnamurthy, and Robert Schapire. Efficient algorithms for adversarial contextual learning. In *International Conference on Machine Learning*, pp. 2159–2168. PMLR, 2016.

Yuchao Tao, Ryan McKenna, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. Benchmarking differentially private synthetic data generation algorithms, 2022. URL `https://arxiv.org/abs/2112.09238`.

Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.

Florian Tramèr, Gautam Kamath, and Nicholas Carlini. Position: Considerations for differentially private learning with large-scale public pretraining. In *Forty-first International Conference on Machine Learning, ICML 2024*, 2024.

Toan V Tran and Li Xiong. Differentially private tabular data synthesis using large language models. *arXiv preprint arXiv:2406.01457*, 2024.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Giuseppe Vietri, Grace Tian, Mark Bun, Thomas Steinke, and Zhiwei Steven Wu. New oracle-efficient algorithms for private synthetic data release, 2020. URL https://arxiv.org/abs/2007.05453.

Giuseppe Vietri, Cedric Archambeau, Sergul Aydore, William Brown, Michael Kearns, Aaron Roth, Ankit Siva, Shuai Tang, and Steven Z Wu. Private synthetic data for multitask learning and marginal queries. *Advances in Neural Information Processing Systems*, 35:18282–18295, 2022.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan, 2019. URL https://arxiv.org/abs/1907.00503.

Mengmeng Yang, Chi-Hung Chi, Kwok-Yan Lam, Jie Feng, Taolin Guo, and Wei Ni. Tabular data synthesis with differential privacy: A survey. *arXiv preprint arXiv:2411.03351*, 2024.

Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021.

Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):1–41, 2017.

Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. {PrivSyn}: Differentially private data synthesis. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 929–946, 2021.

A Appendix

A.1 Existing Work

A.1.1 Existing Algorithms

Broadly, past research on solving DP tabular data synthesis can be divided into two categories: statistical methods and machine learning methods. The exploration of statistical methods started earlier. Shortly after the development of DP, researchers began investigating the problem of data generation under the DP framework. A seminal contribution in this area is MWEM (Hardt et al., 2012), which releases data by minimizing an entropy-regularized loss constructed on marginal queries. A similar approach is DualQuery (Gaboardi et al., 2015), which synthesizes data by iteratively reducing marginal query errors. Another notable line of research involves Bayesian network-based methods, such as PrivBayes (Zhang et al., 2017) and BSG (Bindschaedler et al., 2017), which estimate joint data distributions using low-dimensional marginals. Other methods proposed during this period employ different techniques, such as Copula functions for data generation, as the example demonstrated in (Li et al., 2014).

In 2018, the NIST hosted a challenge about DP data synthesis. Among the competing algorithms, PrivBayes, MST (McKenna et al., 2021), and PrivSyn (Zhang et al., 2021) exhibited superior performance. Both MST and PrivSyn represent datasets using low-dimensional, highly correlated

marginals. Their main difference is that MST synthesizes data using probabilistic graphical models (PGMs) (McKenna et al., 2019), while PrivSyn iteratively updates an initialized dataset during the synthesis process, which is called GUM in their work.

Recently, DP tabular data synthesis has advanced significantly, with several statistical methods emerging. (Cai et al., 2021) introduced PrivMRF, and (McKenna et al., 2022) proposed AIM, both of which dynamically select low-dimensional marginals and employ PGMs for synthesis. Methods such as FEM (Vietri et al., 2020) and RAP/RAP++ (Aydore et al., 2021; Vietri et al., 2022) treat synthesis as an optimization problem, utilizing FTPL (Kalai & Vempala, 2005; Suggala & Netrapalli, 2020; Syrgkanis et al., 2016) and relaxed projection, respectively, to refine initialized datasets using adaptively selected marginals. More recently, (Liu et al., 2023) proposed Private-GSD, which can apply genetic algorithms to iteratively adjust datasets based on any selected marginals.

In addition to statistical methods, machine learning models have been widely explored for DP tabular data synthesis. Early efforts using generative adversarial networks (GANs), such as DP-GAN (Xie et al., 2018), DP-WGAN (Srivastava & Alzantot, 2019), DP-CGAN (Torkzadehmahani et al., 2019), and PATE-GAN (Jordon et al., 2018), demonstrated promise but generally delivered limited performance. More recent methods, including GEM (Liu et al., 2021) and DP-MERF (Harder et al., 2021), represent generative network-based advancements. GEM combines generative networks with adaptive marginal selection mechanisms, while DP-MERF employs random Fourier feature loss to train generative networks. Additionally, TabDDPM (Kotelnikov et al., 2023) leverages diffusion models' representational power to fit target data directly. While TabDDPM was not originally designed for DP, it achieves state-of-the-art performance among non-DP methods and can be adapted for DP synthesis using DP-SGD (Du & Li, 2024). Consequently, we include TabDDPM in our analysis.

Our work focuses on recently proposed methods that are not well compared previously and those representing the current state-of-the-art. Additionally, there are several methods that we do not discuss in detail. For example, (Tran & Xiong, 2024) explores using large language models (LLMs) for data synthesis, showing superior performance compared to many methods. However, LLMs are trained on extensive public datasets. The extensive unknown public datasets potentially overlap sensitive datasets (Tramèr et al., 2024), which may lead to unfair comparisons or break the DP framework, so we exclude it from our work.

A.1.2 Existing Benchmark Work

A.2 Existing Benchmark Works

In addition to the proposed algorithms, there are several benchmark studies (Du & Li, 2024; Hu et al., 2023; Yang et al., 2024; Fan, 2020) that touch the problem of differentially private data synthesis. (Du & Li, 2024) propose systematic assessment metrics for the overall utility evaluation. Their work includes some non-DP methods such as TVAE (Xu et al., 2019) and GreaT (Borisov et al., 2022) but ignores some DP methods like AIM, RAP++ and Private-GSD. (Hu et al., 2023) provide analysis for a wide range of DP synthesis algorithms, not only about tabular data synthesis but also trajectory data and graph data. However, because they involve so many algorithms, they do not do a deeper analysis of these algorithms and lack comprehensive experiments. (Yang et al., 2024) do not delve deep into current algorithms' working principles. Instead, they try to provide a survey of current algorithms and pay attention to their downstream problems, such as distributed data synthesis. (Fan, 2020) specifically focuses on DP generative adversarial networks, which is a subset of DP synthesis methods. In summary, our work differs from previous benchmarks in two key aspects:

- **Unified Framework Proposal**. To ensure the fairness and completeness of algorithm evaluation, we propose a unified algorithm framework (see in Section 2.3). The framework consists of a *data preprocessing module*, a *feature selection module*, and a *data synthesis module*, which serves as the foundation of deep analysis and evaluation.
- **Comprehensive Experiments**. We design systematic experiments to provide deep insights into current algorithms, which involve both end-to-end and module-wise comparisons. While some existing benchmarks (Hu et al., 2023; Yang et al., 2024; Fan, 2020) provide valuable analysis but lack extensive experimental validation, our work combines both detailed analysis and comprehensive experiments.

Table 2: Summary of investigated datasets.

Name	#Records	#Attr	#Num	#Cat	Min/Max Domain
ACSincome (INC) (Ding et al., 2021)	55320	10	2	8	2∼93
ACSemploy (EMP) (Ding et al., 2021)	37881	17	1	16	2∼92
Bank (BK) (Moro et al., 2014)	45211	16	6	10	2∼6024
Higgs-small (HIG) (hig, 2016)	98049	28	28	1	2∼73715
Loan (LN) (loa, 2020)	134658	42	25	17	2∼93995

A.3 Experimental Settings

A.3.1 Datasets

The selection of datasets is crucial in comparison. To make a comprehensive comparison, we would expect that (1) the datasets should vary in record size and attributes' domain sizes to better demonstrate the capacity of methods in synthesizing datasets of different complexity; (2) the proportion of numerical attributes and categorical attributes should be different to better evaluate different preprocessing methods. Therefore, we choose five datasets used in previous work (Liu et al., 2023; McKenna et al., 2022; Zhang et al., 2021; Kotelnikov et al., 2023) as our datasets, which are ACSincome (INC), ACSemploy (EMP), Bank (BK), Higgs-small (HIG) and Loan (LN). The detailed information on these datasets is provided in the appendix, shown in Table 2.

A.3.2 Evaluation Metrics

Various evaluation approaches have been proposed (Zhang et al., 2021; McKenna et al., 2022; Du & Li, 2024). In a nutshell, current evaluation methods are mainly constructed on two basic ideas. The first is to examine the extent to which the generated data can replace the original data set for downstream tasks. The second one is to measure the statistical similarity between the synthetic dataset and the raw dataset. We mainly consider these two categories of evaluation and utilize three metrics: machine learning efficiency, query error, and fidelity error.

Machine Learning Efficiency (higher is better). A widely accepted metric for evaluating generated data is its performance on downstream tasks. A common approach involves training Machine Learning (ML) models on the generated data and assessing their performance on test data. Typically, one column in the dataset is selected as the "label", and a simple ML model is trained to predict the label using the remaining columns as features.

The previous works typically select one or more ML models as downstream tasks. For example, PrivSyn (Zhang et al., 2021) employs an SVM model, TabDDPM (Kotelnikov et al., 2023) uses MLP and CatBoost models, and (Du & Li, 2024) propose eight models for evaluation. However, it's important to note that a large number of models do not necessarily lead to a fairer evaluation. Generally, simpler models have weaker data-fitting capabilities, so their ability to capture data features is limited. Using these simpler models for evaluation can make it harder to reach fair conclusions. In our comparison, we therefore selected four machine learning models known for strong performance across various datasets: MLP, CatBoost, XGBoost, and Random Forest. We use the average F1 score on held-out test data as our metric value.

Another justification is that, for this and the remaining metrics, we use the test dataset instead of the training dataset for evaluation. While both approaches have been employed in previous works, we opt to use test data for evaluation due to the belief that it better reflects an algorithm's generalization ability.

Query Error (lower is better). Making queries (Chatfield, 2018) is a commonly used data analysis technique, which can also be conducted to measure relatively high-dimensional similarity due to its high efficiency. Here, we consider using the 3-way marginal query method used by (Du & Li, 2024; McKenna et al., 2019), which utilizes the statistical ℓ_1 error of frequency query result to reflect the magnitude of the error. Formally, the query error can be expressed as, $\mathbb{E}_{r\in R}\left|q_r(D_{syn}) - q_r(D_{test})\right|$, where q_r refers to the query function, which is a combination of range query (for numerical attributes) and point query (for categorical attributes). $\mathbb{E}$ is the mathe-

matical expectation, and R refers to the set of all 3-way marginals. The better the data generation performs, the smaller this error will be.

Fidelity Error (lower is better). Marginal Fidelity is precise in evaluating low-dimensional similarity, such as average 2-way marginal discrepancy. For example, the work proposed by (Du & Li, 2024) suggests using Wasserstein distance for 2-way marginals to compare the fidelity differences between two distributions, but this method can be infeasible in terms of computation time when dealing with complex data. Alternatively, total variation distance (TVD) can be used for measurement, which has also been utilized in some studies (Tao et al., 2022; Tran & Xiong, 2024). We define the TVD as $\frac{1}{2}\sum_{1\leq i\leq j\leq d}\left|M_{i,j}^{syn}-M_{i,j}^{test}\right|$, where $M_{i,j}^{syn}$ and $M_{i,j}^{test}$ are the real 2-way marginals determined by the synthetic dataset and test dataset respectively.

We also use other metrics in our experiments, such as running time. These metrics are simple enough, so we do not discuss them.

A.4 Algorithm Implementation and Hyperparameters

A.4.1 Overall Implementation

In our experiments, we consider PrivSyn, PrivMRF, RAP++, AIM, Private-GSD, GEM, DP-MERF, and TabDDPM. We do not include RAP in the experiments because RAP++ directly improves upon it. Moreover, notice that Private-GSD is a synthesis algorithm, we use its one-shot 2-way marginal version in the overall evaluation, which is also used in their original work, and pay more attention to its performance in module comparison. Finally, we train TabDDPM under DP-GSD by opacus (Yousefpour et al., 2021). We repeat all evaluations five times and report the average results. The DP parameter δ is set to be 10^{-5} by default. PrivSyn and AIM are executed on CPUs, while the other methods are executed with GPUs.

We can observe that our datasets vary in attributes' domain range, so preprocessing is necessary. We apply uniform discretization and rare category merging for all algorithms as the default aligned preprocessing methods. Moreover, as shown in Table 2, some attributes only contain a few unique values, which are simple enough to handle without additional preprocessing. Applying preprocessing to such attributes is unnecessary and may introduce more errors. Therefore, we preprocess attributes only when their domain size exceeds 100. By default, the number of bins is set to 100, the fixed merging threshold θ to 0.2%, and the privacy budget proportion allocated to the preprocessing step to 10%.

A.4.2 Preprocessing Algorithms

Uniform Binning. The default preprocessing method for numerical attributes is uniform binning. (Dick et al., 2023) use uniform binning in their study. It partitions an attribute's domain into equal-length intervals, relying only on the attribute's domain range and a predefined number of bins. Formally, this method can be expressed as

$$\text{Uniform Bin}(x)=\left\lfloor\frac{x-x_\ell}{h}\right\rfloor,$$

where x_ℓ is the lower bound of the attribute's domain, and h is the length of the uniform interval determined by the bin number. Uniform binning is advantageous because it requires no detailed data information, avoiding the need for additional privacy budget allocation.

Rare Category Merging. Some categorical variables contain a large number of possible values with low counts. This makes computations on them, such as marginal statistics and further model fitting, inefficient. To address this, prior work (Zhang et al., 2021; McKenna et al., 2021) has proposed a 3σ merging strategy, where categories with counts below 3σ (where σ represents the DP noise standard deviation in the counting process) are combined. This approach helps mitigate the impact of noise on the counts of individual categories. However, this method has limitations: when we have plenty of privacy budget, 3σ could be a small value. We cannot reduce the attributes' domain size if we still use 3σ as the threshold for combining.

In response to these limitations, we improve the 3σ merging method by applying a dual merging threshold. For target attributes, categories with counts below a fixed threshold $n\theta$ or the privacy

Algorithm 1: DP Rare Category Merge

Input: dataset D, merge threshold parameter θ, unique value threshold β, DP parameter ρ_2
Output: preprocessed dataset D

$V_c \leftarrow$ categorical variables with domain size $\geq \beta$;
$\rho' \leftarrow \rho_2/|V_c|$;
for $j \in V_c$ **do**
  $b \leftarrow$ 1-way marginal of attribute A_j;
  $\sigma \leftarrow \sqrt{\frac{1}{2\rho'}}$; ▷ Noise scale
  $\hat{b} = b + \mathcal{N}\left(0, \sigma^2\right)$;
  for $i = 1 : |\hat{b}|$ **do**
    $\theta' \leftarrow \max\left\{\theta \cdot \sum \hat{b},\ 3\sigma\right\}$; ▷ Merging threshold
    if $\hat{b}[i] < \theta'$ **then**
      replace $\hat{b}[i]$ with the rare encoding value;
    end
  end
end
return D

budget-aware threshold 3σ are replaced with a rare encoding value. Here n is the number of records in the dataset and θ is the threshold parameter. By introducing a fixed threshold, we can avoid the case when σ is too small to reduce the attribute's domain complexity. The detailed description of this algorithm is shown in Algorithm 1. For each attribute that needs preprocessing, we equally divide the privacy budget and use it to determine those categories whose frequency is lower than the threshold. These categories will be replaced by the same encoding category. The privacy guarantee can be formalized in the following lemma.

Lemma 1 *For any $\alpha > 1$, Algorithm 1 satisfies $(\alpha, \alpha\rho_2)$-Rényi differential privacy.*

The proof of this lemma can be easily obtained by the property of the Gaussian mechanism. Referring to (Mironov, 2017), we know that adding Gaussian noise with $\sigma = \sqrt{\frac{1}{2\rho'}}$ satisfying $(\alpha, \alpha\rho')$-Rényi differential privacy for any $\alpha > 1$. Then by combining the fact that $\rho' = \rho_2/|V_c|$ and the composition property of RDP, we have that the total algorithm satisfies $(\alpha, \alpha\rho_2)$-Rényi differential privacy for any $\alpha > 1$.

A.4.3 Hyperparameters for Full Algorithms

We list the algorithms' hyperparameters in Section 3.1. From here on out, unless otherwise specified, INC refers to the ACSincome dataset; EMP refers to the ACSemploy dataset; BK refers to the Bank dataset; HIG refers to the Higgs-small dataset; LN refers to the Loan dataset.

Table 3: PrivSyn Hyperparameters

Hyperparameter	**INC**	**EMP**	**BK**	**HIG**	**LN**
Consistent iteration	501	501	501	501	501
Max update iteration	50	50	50	50	50

Table 4: AIM Hyperparameters

Hyperparameter	**INC**	**EMP**	**BK**	**HIG**	**LN**
Max model size	100	100	100	100	100
Max iteration	1000	1000	1000	1000	1000
Max marginal size	$2.5e+5$	$2.5e+5$	$2.5e+5$	$2.5e+5$	$2.5e+5$

Table 5: Private-GSD Hyperparameters

Hyperparameter	**INC**	**EMP**	**BK**	**HIG**	**LN**
Mutation number	50	50	50	50	50
Crossover number	50	50	50	50	50
Upsample number	$1e+5$	$1e+5$	$1e+5$	$1e+5$	$1e+5$
Genetic iteration	$1e+6$	$1e+6$	$1e+6$	$1e+6$	$1e+6$

Table 6: PrivMRF Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Graph parameter θ	6	6	6	6	6
Sample size k	400	400	400	400	400
Estimation iteration	3000	3000	3000	3000	3000
Size penalty	$1e-8$	$1e-8$	$1e-8$	$1e-8$	$1e-8$
Max marginal dimension	6	6	6	6	6
Max clique size	$1e+7$	$1e+7$	$1e+7$	$1e+7$	$1e+7$

Table 7: RAP++ Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Cat optimization rate	$3e-3$	$3e-3$	$3e-3$	$3e-3$	$3e-3$
Num optimization rate	$6e-3$	$6e-3$	$6e-3$	$6e-3$	$6e-3$
Top q	5	5	5	5	5
Cat optimization step	1	1	1	1	1
Num optimization step	3	3	3	3	3
Upsample rate	10	10	20	20	40
Random projection number	$2e+6$	$2e+6$	$2e+6$	$2e+6$	$2e+6$

Table 8: GEM Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Synthesis size	1024	1024	1024	1024	1024
Learning rate	$1e-3$	$1e-3$	$1e-3$	$1e-3$	$1e-3$
Max iteration	500	500	500	500	500
Max selection round	50	85	80	140	210

Table 9: DP-MERF Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Mini batch rate	$5e-2$	$5e-2$	$5e-2$	$5e-2$	$5e-2$
Epoch number	$1e+3$	$1e+3$	$1e+3$	$1e+3$	$1e+3$
Learning rate	$1e-2$	$1e-2$	$1e-2$	$1e-2$	$1e-2$
Random feature dimension	$2e+3$	$2e+3$	$2e+3$	$2e+3$	$2e+3$

Table 10: TabDDPM Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Epoch number	50	100	100	100	100
Batch size	512	512	512	1024	1024
Learning rate	$2e-2$	$1e-2$	$5e-3$	$5e-2$	$5e-4$
Diffusion steps	100	100	100	1000	100
Layer number	2	2	2	2	2
Layer dimension	256	256	256	256	1024

A.4.4 Hyperparameters for Different Feature Selection Algorithms

The PrivMRF and AIM selection algorithms are set to completely the same as the original work in PrivMRF and AIM, respectively. Therefore, we omit the description of them here and provide the detailed hyperparameter setting of RAP++ selection and PrivSyn selection.

Table 11: RAP++ Selection Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Top q	3	3	3	3	3
Selection step	4	6	6	7	10
Selection budget rate	0.5	0.5	0.5	0.5	0.5
Marginal budget rate	0.5	0.5	0.5	0.5	0.5

Table 12: PrivSyn Selection Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Selection budget rate	0.1	0.1	0.1	0.1	0.1
1-way marginal budget rate	0.1	0.1	0.1	0.1	0.1
2-way marginal budget rate	0.8	0.8	0.8	0.8	0.8

A.4.5 Hyperparameters for Different Synthesis Algorithms

Most synthesis algorithms we used are set to be the same as their original works, while relaxed projection and generative network methods need hyperparameter tuning to guarantee performance.

Table 13: Relaxed Projection Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Optimization rate	$5e-3$	$5e-3$	$5e-3$	$5e-3$	$5e-3$
Optimization step	100	170	160	280	420
Random projection number	$2e+6$	$2e+6$	$2e+6$	$2e+6$	$2e+6$

Table 14: Generative Network Hyperparameters

Hyperparameter	INC	EMP	BK	HIG	LN
Learning rate	$1e-3$	$1e-3$	$1e-3$	$1e-3$	$1e-3$
Synthesis size	1024	1024	1024	1024	1024
Training iteration	50	50	100	100	1500

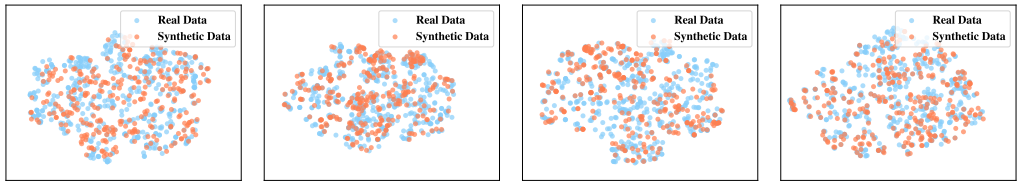
A.5 Supplementary Experiment Results

A.5.1 T-SNE Plots

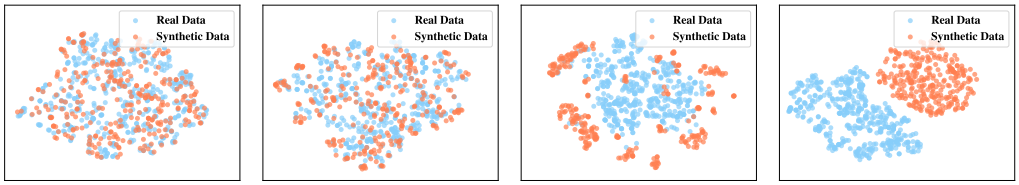
We draw t-SNE plots of different algorithms on the Bank dataset. By default, we set $\varepsilon = 1.0$. Generally, synthetic data distribution by PrivSyn, PrivMRF, RAP++, AIM, GEM, and Private-GSD align better with real data distribution. On the contrary, DP-MERF and TabDDPM's synthetic data distribution have some significant gap with real data distribution.

A.5.2 Running Time

We present the running times of these algorithms in Table 15 (also with some supplementary metrics for machine learning efficiency). On average, most machine learning-based methods are time-efficient (partially because they use GPU in model-fitting steps). Among these algorithms, RAP++,



(a) PrivSyn (b) PrivMRF (c) RAP++ (d) AIM



(e) Privtae-GSD (f) GEM (g) DP-MERF (h) TabDDPM

Figure 5: T-SNE scatter plots of synthesis results on Bank dataset under $\varepsilon = 1.0$

Table 15: Supplementary overall results of synthetic data under different methods. ML AUC and ML Accuracy are metrics obtained by downstream ML tasks. Running time is the total average execution time of the algorithm. Because Loan dataset is a multi-classification problem, thus it does not have AUC result.

Dataset	**ACSincome**			**ACSemploy**			**Bank**			**Higgs-small**			**Loan**		
ML AUC	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
PrivSyn	0.53	0.50	0.51	0.47	0.47	0.42	0.44	0.52	0.49	0.50	0.50	0.50	-	-	-
PrivMRF	0.82	0.87	0.87	0.80	0.86	0.89	0.71	0.90	0.92	0.53	0.71	0.70	-	-	-
RAP++	0.75	0.82	0.85	0.81	0.84	0.87	0.75	0.85	0.88	0.54	0.56	0.56	-	-	-
AIM	0.85	0.87	0.87	0.85	0.88	0.88	0.87	0.89	0.91	0.68	0.72	0.74	-	-	-
Private-GSD	0.85	0.85	0.85	0.78	0.80	0.79	0.68	0.67	0.67	0.52	0.52	0.52	-	-	-
GEM	0.77	0.73	0.72	0.75	0.77	0.77	0.59	0.68	0.69	0.55	0.57	0.59	-	-	-
DP-MERF	0.75	0.78	0.79	0.73	0.78	0.74	0.72	0.64	0.65	0.54	0.60	0.60	-	-	-
TabDDPM	0.54	0.49	0.53	0.56	0.56	0.55	0.48	0.51	0.45	0.51	0.53	0.53	-	-	-
ML Accuracy	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
PrivSyn	0.59	0.58	0.59	0.52	0.49	0.49	0.88	0.88	0.88	0.53	0.52	0.52	0.54	0.54	0.54
PrivMRF	0.75	0.79	0.79	0.72	0.79	0.81	0.89	0.90	0.90	0.53	0.65	0.64	0.76	0.75	0.76
RAP++	0.68	0.75	0.78	0.74	0.78	0.80	0.85	0.88	0.89	0.53	0.55	0.55	0.62	0.64	0.65
AIM	0.78	0.79	0.79	0.78	0.80	0.81	0.90	0.90	0.90	0.63	0.65	0.67	0.75	0.75	0.75
Private-GSD	0.77	0.77	0.78	0.71	0.73	0.71	0.87	0.87	0.88	0.51	0.51	0.52	0.54	0.54	0.54
GEM	0.71	0.69	0.66	0.68	0.70	0.69	0.86	0.86	0.87	0.55	0.55	0.56	0.71	0.68	0.72
DP-MERF	0.69	0.70	0.72	0.64	0.71	0.67	0.84	0.79	0.74	0.53	0.58	0.58	0.35	0.37	0.37
TabDDPM	0.57	0.59	0.59	0.55	0.54	0.53	0.88	0.88	0.88	0.50	0.53	0.51	0.55	0.55	0.55
Running Time (min)	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
PrivSyn	0.28	0.16	0.18	0.30	0.25	0.34	0.57	0.53	0.55	5.39	5.95	4.54	13.29	14.04	13.44
PrivMRF	1.26	0.89	1.40	5.59	5.80	4.76	3.06	4.48	3.24	5.06	7.10	6.25	7.12	13.34	12.62
RAP++	25.86	24.95	25.72	26.28	27.04	28.30	24.40	23.94	23.27	37.79	36.66	35.88	544.25	2348.59	1761.60
AIM	1.96	6.29	126.89	5.37	10.59	443.70	3.60	10.23	23.57	12.62	18.46	184.57	31.03	187.27	645.13
Private-GSD	22.33	24.86	26.17	10.89	11.12	11.11	19.38	20.01	19.99	54.35	57.49	57.93	304.42	306.04	311.63
GEM	0.26	0.10	0.09	0.12	0.12	0.12	0.27	0.25	0.25	6.05	6.09	10.75	9.82	9.47	9.82
DP-MERF	0.40	0.07	0.06	0.30	0.06	0.06	0.09	0.06	0.06	0.09	0.06	0.06	0.39	0.83	0.34
TabDDPM	4.34	4.12	3.73	4.52	4.50	13.31	8.25	4.40	3.87	4.74	6.34	4.78	27.54	31.93	33.34

Table 16: Influences of preprocessing on different datasets. By default, the results are obtained under the setting that $\varepsilon = 1.0$ and 10% of the budget is used for preprocessing.

Preprocessing Method	**Marginal Size**			**ML Efficiency**			**Query Error**			**Fidelity Error**		
	Bank	Higgs-small	Loan	Bank	Higgs-small	Loan	Bank	Higgs-small	Loan	Bank	Higgs-small	Loan
Raw	$1.05e+5$	$4.24e+8$	$2.88e+8$	0.76	0.72	0.54	0.001	0.001	0.001	0.003	0.001	0.001
Numerical preprocessing	$6.48e+2$	$5.05e+3$	$1.14e+6$	0.76	0.71	0.53	0.002	0.002	0.003	0.006	0.002	0.003
Categorical Preprocessing	$1.05e+5$	$4.24e+8$	$2.31e+8$	0.76	0.72	0.54	0.001	0.001	0.001	0.003	0.001	0.001
Full Preprocessing	$6.48e+2$	$5.05e+3$	$3.14e+3$	0.76	0.71	0.53	0.002	0.002	0.003	0.006	0.002	0.003

AIM, and Private-GSD require longer execution times. The lower efficiency of RAP++ and AIM can be attributed to the multi-rounds of feature fitting required in each adaptive feature selection iteration. In addition, running AIM on CPUs will also influence its efficiency. Moreover, because a higher budget allows AIM to select more marginals, we can observe that AIM's running time increases significantly with the increase of privacy parameter. In addition, Private-GSD requires many rounds to adjust the initial data, resulting in slower execution speeds.

Table 17: Results of synthetic data under different feature selection methods. By default, we use PGM as the synthesis method. In this table, "-" means unable to execute due to time or memory limitation.

Dataset	**ACSincome**			**ACSemploy**			**Bank**			**Higgs-small**			**Loan**		
ML efficiency	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
PrivSyn selection	0.67	0.65	0.68	0.67	0.80	0.75	–	–	–	–	–	–	–	–	–
PrivMRF selection	0.73	0.78	0.78	0.81	0.80	0.81	0.62	0.70	0.71	0.50	0.64	0.64	0.52	0.52	0.52
RAP++ selection	0.62	0.74	0.74	0.74	0.78	0.79	0.51	0.51	0.47	0.49	0.49	0.50	0.30	0.26	0.26
AIM selection	0.76	0.78	0.78	0.78	0.80	0.81	0.67	0.71	0.70	0.63	0.65	0.67	0.52	0.52	0.52
Query Error	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
PrivSyn selection	0.003	0.001	0.001	0.003	0.002	0.003	–	–	–	–	–	–	–	–	–
PrivMRF selection	0.002	0.001	0.001	0.003	0.002	0.002	0.005	0.003	0.003	0.005	0.003	0.003	0.005	0.005	0.004
RAP++ selection	0.006	0.003	0.002	0.008	0.005	0.004	0.012	0.005	0.003	0.047	0.016	0.006	0.015	0.009	0.008
AIM selection	0.002	0.001	0.001	0.003	0.002	0.001	0.007	0.002	0.002	0.005	0.003	0.003	0.005	0.005	0.004
Fidelity Error	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
PrivSyn selection	0.14	0.08	0.07	0.07	0.04	0.04	–	–	–	–	–	–	–	–	–
PrivMRF selection	0.11	0.07	0.05	0.07	0.04	0.03	0.13	0.06	0.04	0.36	0.19	0.19	0.31	0.24	0.23
RAP++ selection	0.17	0.09	0.08	0.04	0.04	0.04	0.14	0.09	0.08	0.42	0.23	0.17	0.32	0.26	0.25
AIM selection	0.09	0.06	0.05	0.05	0.03	0.02	0.11	0.09	0.06	0.21	0.16	0.16	0.35	0.32	0.29

Table 18: Results of synthetic data under different synthesis methods. By default, we use PrivSyn's InDif selection as the selection method. In this table, "-" means unable to execute due to time or memory limitation.

Dataset	**ACSincome**			**ACSemploy**			**Bank**			**Higgs-small**			**Loan**		
ML efficiency	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
GUM	0.39	0.40	0.42	0.45	0.45	0.40	0.47	0.47	0.47	0.40	0.43	0.43	0.25	0.26	0.26
PGM	0.76	0.64	0.67	0.68	0.80	0.75	–	–	–	–	–	–	–	–	–
Relaxed Projection	0.39	0.59	0.58	0.51	0.69	0.67	0.47	0.47	0.47	0.47	0.44	0.42	0.25	0.25	0.25
Genetic Algorithm	0.67	0.63	0.58	0.62	0.58	0.62	0.61	0.57	0.52	0.59	0.58	0.62	0.35	0.36	0.26
Generative Network	0.74	0.77	0.76	0.72	0.70	0.69	0.66	0.63	0.62	0.63	0.53	0.59	0.46	0.41	0.36
Query Error	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
GUM	0.003	0.002	0.002	0.006	0.004	0.004	0.007	0.004	0.003	0.009	0.004	0.003	0.006	0.005	0.004
PGM	0.003	0.001	0.001	0.003	0.002	0.003	–	–	–	–	–	–	–	–	–
Relaxed Projection	0.040	0.028	0.023	0.081	0.025	0.022	0.047	0.013	0.014	0.026	0.013	0.012	0.020	0.009	0.008
Genetic Algorithm	0.053	0.047	0.050	0.039	0.040	0.036	0.009	0.005	0.003	0.027	0.018	0.016	0.039	0.037	0.039
Generative Network	0.002	0.002	0.002	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.004	0.004
Fidelity Error	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$	$\varepsilon = 0.2$	$\varepsilon = 1$	$\varepsilon = 5$
GUM	0.15	0.12	0.12	0.12	0.09	0.09	0.24	0.10	0.21	0.29	0.20	0.20	0.34	0.35	0.36
PGM	0.13	0.08	0.07	0.07	0.04	0.04	–	–	–	–	–	–	–	–	–
Relaxed Projection	0.61	0.52	0.41	0.51	0.26	0.21	0.41	0.16	0.16	0.31	0.22	0.21	0.41	0.29	0.28
Genetic Algorithm	0.59	0.59	0.56	0.30	0.30	0.30	0.13	0.09	0.08	0.35	0.28	0.27	0.56	0.55	0.55
Generative Network	0.08	0.08	0.08	0.06	0.07	0.06	0.16	0.16	0.15	0.24	0.23	0.23	0.47	0.38	0.27

A.5.3 Detailed Results of Preprocessing Influence

The detailed results of different preprocessing methods (used to plot Figure 2) are shown in Table 16. Similar to other experiments, these metrics are calculated by comparing preprocessed datasets with test datasets to demonstrate the error caused by preprocessing.

A.5.4 Detailed Results of Deconstruction Experiment

The detailed results of different preprocessing methods (used to plot Figure 3 and Figure 4) are shown in Table 17 andTable 18. Here, there are some "-" in the table. This is because PrivSyn tends to select as many marginals as possible, which will form large cliques. This will cause the size of the graphical model to be too large, requiring extremely large memory.