On the Usefulness of Synthetic Tabular Data Generation

Dionysis Manousakas [1] **Sergül Aydöre** [1]

Abstract

Despite recent advances in synthetic data generation, the scientific community still lacks a unified consensus on its usefulness. It is commonly believed that synthetic data can be used for both data exchange and boosting machine learning (ML) training. Privacy-preserving synthetic data generation can accelerate data exchange for downstream tasks, but there is not enough evidence to show how or why synthetic data can boost ML training. In this study, we benchmarked ML performance using synthetic tabular data for four use cases: *data sharing*, *data augmentation*, *class balancing*, and *data summarization*. We observed marginal improvements for the balancing use case on some datasets. However, we conclude that there is not enough evidence to claim that synthetic tabular data is useful for ML training.

1. Introduction

Generative modeling is a powerful technique that can be used to create synthetic data that mimics the properties of a given population of real data. This can be a valuable tool for machine learning applications, as it can help to improve predictive ability and fairness.

When working with scarce or imbalanced datasets (Kim et al., 2022; Ai et al., 2023), generative modeling can be used to augment the data by creating synthetic data points that fill in the gaps. This can help to improve the performance of machine learning models, as they will have more data to train on.

In addition to improving predictive ability, generative modeling can also be used to improve the fairness of machine learning models. This is because synthetic data can be created to be more representative of the population as a whole, which can help to reduce bias in the models. For example, a study by van Breugel et al. (2021) found that augmenting a dataset with synthetic data can help to improve the fairness of a machine learning model for predicting recidivism rates.

In this study, our focus is tabular data, which poses unique challenges for generative modeling, including:

- Lack of local structure: Tabular data does not have the same local structure as image, audio, or text data. This makes it difficult for generative models to learn the relationships between different features.
- Co-existence of categorical and numerical features: Tabular data often contains a mix of categorical and numerical features. This can make it difficult for generative models to learn a joint distribution over all of the features.
- Feature missingness: Tabular data often contains missing values. This can make it difficult for generative models to learn a complete distribution over all of the features.
- High scarcity: Tabular data is often scarce. This means that there is not enough data to train a generative model.

As a result of these challenges, advances in generative models for tabular data have lagged behind advances in image, audio, and text data (Kim et al., 2023; Kotelnikov et al., 2022). In order to address these challenges, researchers have developed tailored synthetic data quality metrics. These metrics can be used to evaluate the quality of synthetic tabular data, and to help ensure that the data is suitable for use in machine learning applications.

In this study, we conduct a range of experiments to evaluate the applicability of synthetically generated data for generic downstream tabular learning tasks. We focus on four axes of practical interest:

- *Data sharing / Synthetic data quality*: We evaluate the downstream performance of models trained on synthetic data and compare it to the performance of models trained on the original data distribution. For synthetic data to be useful, machine learning development performed on synthetic data should lead to the same conclusions as if it were carried out on the real data. The train-on-synthetic and test-on-real (TS-TR) paradigm is the default experiment for evaluating the usefulness

[1]Amazon AWS AI/ML. Correspondence to: Dionysis Manousakas <dionman@amazon.co.uk >.

of generative modeling methods for data sharing. Here are some additional details about the TS-TR paradigm:

- In the TS-TR paradigm, a generative model is trained on a dataset of real data.
- The generative model is then used to generate synthetic data.
- A machine learning model is trained on the synthetic data.
- The performance of the machine learning model is evaluated on a hold-out dataset of real data.

The TS-TR paradigm is a useful tool for evaluating the usefulness of generative modeling methods for data sharing. However, it is important to note that the TS-TR paradigm does not provide formal privacy guarantees. If privacy is a concern, additional noise may need to be added to the synthetic data as in Vietri et al. (2022).

- *Summarization*: In this setting, we evaluate whether we can use synthetic data to represent the original distribution with fewer samples. To do this, we replace a given data population with a controllable, smaller number of synthetic data points sampled from a generative model that has been trained on the full real dataset. We then use the synthetic data for downstream learning tasks. In contrast to the *data sharing* study outlined above, where we fixed the volume of synthetic data to equal the real data used for the generative model training, in the TS-TR setting of this experiment, we focus on the trade-off between the number of synthetic data points and the achievable downstream ML efficiency.

- *Augmentation*: To address the issue of data scarcity, original training data can be augmented with synthetic data to create a richer training dataset. In this setting, we augment real data populations with synthetically generated data before performing downstream learning tasks. We then optimize the end learner on the augmented training dataset and evaluate it on real test data. We call this approach train-on-augmented, test-on-real (TA-TR).

- *Class balancing*: For classification tasks, we can sample synthetic data conditionally from the minority categories of a given dataset to reduce or eliminate class imbalance prior to downstream training. We then use the balanced version of the data as a training dataset for the downstream classifier (TA-TR). This setting differs from the previous one in that we sample synthetic data conditionally on the minority class.

For the synthetic data generation, we consider two generative modelling approaches (cf. Section 2.2): (i) deep generative models (DGMs) trained from scratch using a real data population at hand, and (ii) fine-tuned versions of pre-trained large language models (LLMs) that incorporate prior information beyond the real data at hand. These experiments aim to unveil if we can achieve an equivalent or better representation of the training data distribution for general, downstream model-agnostic learning purposes, via leveraging synthetic samples from generative models. We also examine whether the synthetic samples have the capacity to efficiently compress statistical information from the training data population without further guidance, allowing relying on a smaller sample size for downstream tasks, without statistical quality loss. This could unlock the potential of accelerating downstream learning tasks.

With the exception of the synthetic data quality evaluation, the rest of our experiments have a smaller footprint in the literature. Studies in deep generative model based data balancing have reported only weak improvements to lightweight baselines in the tabular domain (Elor & Averbuch-Elor, 2022), with only recently proposed specialised generative architectures being able to more confidently outperform SMOTE (Kim et al., 2022). In addition, we are unaware of experiments with generative models addressing tabular data summarization, while an augmentation setting similar to ours has been considered in Liu et al. (2023) reporting no significant downstream boosting.

2. Benchmarking

Below we give an overview of the components involved in our benchmarking setup. These include a set of generative models (Section 2.1) that enable us to sample synthetic datapoints w/ and w/o conditioning, a set of downstream learners (Section 2.2) which correspond to the end learning task for the data distribution at hand, and some public access tabular datasets (Section 2.3) that we use for our evaluation. We conclude this section with further details on our experimental setup (Section 2.4).

2.1. Generative models

We include the following state-of-the-art tabular data generation methods in our experimentation.

Conditional Tabular GANs (CTGANs) (Xu et al., 2019). An adaptation of the original GAN architecture (Goodfellow et al., 2014) with two main modifications for handling tabular datasets: (i) an elaborate normalization scheme for multimodal features, and (ii) a training-by-sampling paradigm for better learning of under-represented categories. Training-by-sampling means that a random column is being selected and conditioned on during each training iteration of the GAN. We overload this notation, referring with the term **Conditional CTGAN** to a CTGAN model that is trained to learn the conditional distribution $p(x|y)$ instead, where y is

Table 1. Dataset statistics and task details.

Dataset	Domain	#Samples	#Num	#Cat	Task	#Classes	Imbalance $\left(\frac{\text{\#Majority pts.}}{\text{\#Minority pts.}}\right)$
adult	Social	32,561	6	8	Classification	2	3.15
churn	Marketing	954	2	4	Classification	2	3.26
sick	Medical	3,772	7	22	Classification	2	11.96
heloc	Financial	9,871	21	2	Classification	2	1.1
california	Real Estate	20,640	8	0	Regression	-	-

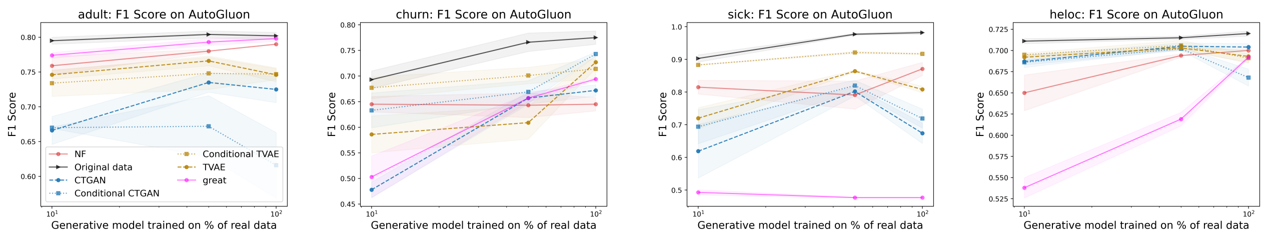


Figure 1. (*Synthetic data quality experiment*) Predictive performance of a downstream AutoGluon classifier, which uses synthetic training data sampled from generative models that have been trained on increasing *subsets* of the real data. The number of synthetic training datapoints increases along the x-axis, matching the number of real data used for training the generative model.

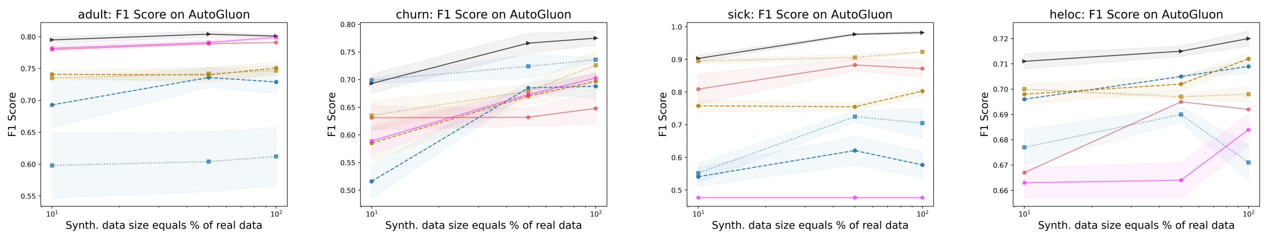


Figure 2. (*Data summarization experiment*) Predictive performance of a downstream AutoGluon classifier which uses synthetic training data sampled from a generative model that has been trained on the *full* real dataset. The number of synthetic training data increases along the x-axis.

a target variable.[1] As in Mirza & Osindero (2014), training in the case of Conditional CTGAN differs from the original training-by-sampling paradigm in that the same conditional column y will be used throughout training. The latter allows conditional sampling on the target variable for the class balancing use case, as at sampling time it enables passing explicitly the information of the desired class y and using the CTGAN to sample $x \sim p(x|y)$, to jointly generate a synthetic sample (x, y). Moreover, it makes use of factorization of the joint that is based on the downstream learning task.

Tabular Variational AutoEncoders (TVAEs) (Xu et al., 2019). An adaptation of the original VAE architecture (Kingma & Welling, 2014) sharing the pre-processing steps of CTGAN. Similarly to CTGANs, we also consider **Conditional TVAEs** that model directly $p(x|y)$, where y is the corresponding target variable.

[1] as opposed to modeling $p(x, y)$ which is the original learning goal of CTGAN

Normalizing Flows (NFs) . A normalizing flow (Papamakarios et al., 2021) is a generative model that transforms a simple distribution to a more complex one by applying a series of invertible transformations. These transformations gradually distort the simple distribution to match the target distribution of the data. By optimizing the parameters of the flow through maximum likelihood estimation, the model learns to generate new samples from the learned distribution and estimate the likelihoods of observed data points accurately. In our tabular learning experiments, we use the method proposed in Durkan et al. (2019) with standard one-hot encoding for categorical features. We only use normalizing flows to learn the data joint probability, hence omit this model from the balancing use case that requires conditional sampling.

Fine-tuned pre-trained Large Language Models (GReaT) . We use the method proposed in Borisov et al. (2023). Training consists of (i) converting the tabular datapoints into sentences via a simple parsing scheme, and (ii) fine-tuning an existing large-language model on the data. We rely on tuning the *distill-GPT2* model (Sanh et al., 2019) on our tabular datapoints at hand for each benchmark (introduced in the original paper as *distil-GReaT*). We adopt the key design choices selected in Borisov et al. (2023), e.g. a perturbations scheme on the extracted sentences. GReaT uses auto-regressive neural networks implementing a transformer decoder-only architecture (Vaswani et al., 2017), which along with the perturbation scheme, enable arbitrary sampling on the datapoint features (including conditioning on the target variable, which is used in the balance use case experiments).

SMOTE (Chawla et al., 2002) A non-parametric method addressing class imbalance in datasets. It works by generating synthetic samples for the minority class by interpolating between existing minority class instances. The algorithm selects a minority class sample and finds its k nearest neighbors. It then randomly selects one or more neighbors and creates new synthetic samples along the line segments joining the original sample and its neighbors. This process helps to increase the representation of the minority class, balancing the class distribution and improving the model ability to learn from the minority class instances.

Original A baseline that subsamples true datapoints uniformly at random in the absence of augmentation. Random sampling controls for the varying sample size effects, capturing the expected performance on a specified volume of real training data w/o interventions via generative models.

For our experiments with DGMs we relied on the implementations of the `synthcity` package (Qian et al., 2023), for GReaT we used the original implementation of the authors `be_great` (Borisov et al., 2023), while for SMOTE we used the implementation of the model for mixed numeric and categorical data from Lemaître et al. (2017).

2.2. Downstream models and metrics

We are interested in measuring the extra downstream utility induced via generative models for broad practical use cases. Hence, to make our evaluation agnostic to a specific category of downstream ML models, for *classification* tasks we consider two powerful tabular classifiers, namely XGBoost (XGB) (Chen & Guestrin, 2016), and AutoGluon (Erickson et al., 2020). For *regression* tasks, we evaluate against a linear regression model (LinReg), a multilayer perceptron (MLPRegressor) and AutoGluon. We present the metrics obtained on AutoGluon in the main part, deferring the remaining models to Appendix B. We keep F1-Score and RMSE as the evaluation metrics respectively for classification and regression.

2.3. Datasets

We give a short overview of the datasets used in our experiments in Table 1. We rely on publicly available datasets (cf. Table 4) comprised of 4 datasets for classification experiments, and 1 for regression, with sizes varying from less than 1k datapoints to more than 30k.

2.4. Experimental setup

We use a fixed split of training and testing datapoints with a $80/20$ percent scheme throughout the entire experiment. For each experiment we repeat the training/fine-tuning of the corresponding generative model across 3 independent trials, and present the obtained mean and standard error of the computed metrics. We make use of a fixed set of training hyperparameters across datasets for each generative model. We apply hyperpameter optimization (HPO) independently per trial for each combination of use case and model over downstream learning. For HPO we use asynchronous random search. Please refer to Appendix C for details on the considered hyperparameter search spaces. As the number of nearest neighbors used to construct artificial minority samples for SMOTE we use $k = 5$ throughout. In the case of GReaT we fine-tuned the distil-GPT model for a total of 200 epochs for each of our datasets, and used the default hyperparameters for training and sampling. For DGMs we trained up to a maximum of 5,000 epochs using early-stopping with a patience of 500 epochs without improvement on a validation score.

3. Results

3.1. Quality of Synthetic Data

In this part of the experiment, we aim to evaluate the statistical quality of the synthetic samples for downstream learning tasks. For this purpose, we remove the original datapoints prior to training the downstream model, and replace them with an *equal volume of synthetic datapoints*. To capture the effects of varying sample sizes of the true data, we report predictive performance for the described TS-TR setup, considering generative models trained on increasing fractions of the original training population (10, 50 and 100%). The obtained downstream performance is shown in Figures 1 and 4. As a general remark, training on the available original data outperforms training on the synthetic samples for all tested generative models. Regarding the latter, we observed that the relative ordering of the downstream efficacy of the compared DGMs does not seem to change much across the varying dataset size regimes, while the gap between synthetic and real data is closing as we use more datapoints to

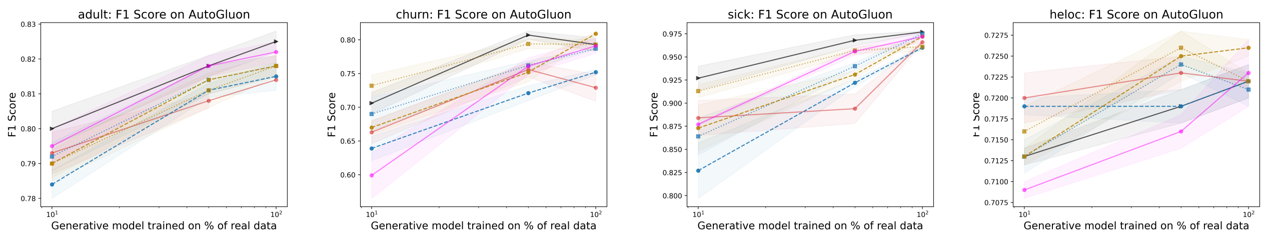
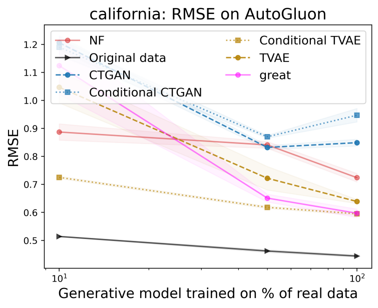
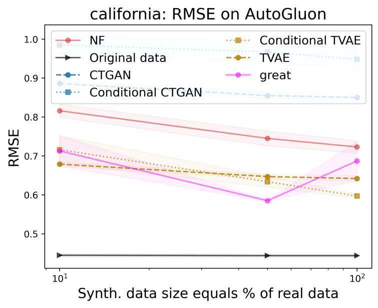


Figure 3. (*Data augmentation experiment*) Predictive performance of a downstream AutoGluon classifier which uses increasing subsets of real data *augmented* with equal-sized volumes of synthetic data.





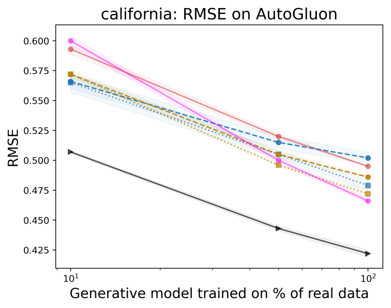


Figure 4. Synthetic data quality experiment for regression

Figure 5. Data summarization experiment for regression.

Figure 6. Data augmentation experiment for regression.

train the generative model. When trained on the full dataset, conditional TVAE maintains the highest rank among DGMs across the datasets. For large datasets with interpretable feature semantics (e.g. adult, see Table 3) we eventually obtain higher quality using the pre-trained LLM. In the case of datasets with higher dimensionality, long contexts of specialised terms or abbreviated feature names, such as sick and heloc, GReaT seems to require fine-tuning on more training examples in order to eventually provide synthetic data quality at the same levels with DGMs.

3.2. Summarization

Here we investigate if sampling from generative models trained on the full original data distribution inherently enables more efficient compression of the statistical information of the original data for the downstream task, compared to extracting random subsets of real datapoints. As in the previous section, we replace the original datapoints entirely with synthetic samples with the help of a generator; however now the generator has been trained on the full real data, and we use volumes of *synthetic data with varying size* corresponding to 10, 50 and 100% of the true data population. As we observe in Figures 2 and 5, the achievable downstream quality is generally lower compared to equal volumes of real datapoints, and marginally higher compared to the corresponding points in Figures 1 and 4, as now the generative models have seen more training examples before producing the same amount of synthetic samples. This hints that sampling from these generative models does not significantly reduce the redundancy in the resulting population to lower levels compared to the original data; ML efficiency seems to be following slightly flatter curves for the synthetic data training sets, and saturates at the same levels with the plots of the synthetic data quality experiment (Section 3.1) for synthetic populations with the same volume as the real data.

3.3. Augmentation

In this experiment, we train our generative models on increasing subsets of the original data population, comprising 10, 50 and 100% of the full data, and complement with equal volumes of synthetic data, sampled from a trained generative model that has seen only the subset. Finally, *we train the downstream classifier on the augmented volume of real and synthetic datapoints*. This setting aims to simulate situations with data scarcity, and provide empirical evidence addressing whether augmenting with generative models can benefit downstream predictive accuracy. We present the ML efficacy achieved with this augmentation strategy for classification and regression tasks in Figures 3 and 6. We observe that augmentations via generative models were not capable to consistently improve the performance that we got when constraining the downstream training data only among the original datapoints.

	Original data	SMOTE	Conditional CTGAN	Conditional TVAE	GReaT
			F1 Score		
adult	0.802±0.009	0.809±0.008	0.808±0.009	**0.812±0.008**	0.805±0.006
churn	0.779±0.023	**0.808±0.011**	0.741±0.003	0.771±0.02	0.777±0.02
sick	**0.982±0.009**	0.974±0.007	0.917±0.013	0.921±0.015	0.977±0.003
heloc	0.721±0.004	0.717±0.003	**0.722±0.003**	0.721±0.002	0.72±0.004
			ROC AUC		
adult	0.908±0.005	0.906±0.004	0.91±0.004	**0.913±0.003**	0.908±0.003
churn	**0.934±0.008**	0.917±0.01	0.899±0.005	0.913±0.013	0.915±0.008
sick	**0.999±0.0**	**0.999±0.001**	0.994±0.001	**0.999±0.001**	**0.999±0.0**
heloc	0.787±0.005	0.784±0.005	0.786±0.005	**0.791±0.006**	0.787±0.005

Table 2. (*Data balancing experiment*) Predictive metricsevaluated on a downstream AutoGluon classifier that uses an rebalanced augmented version of the dataset via samples from the generative models.

3.4. Class balancing

In our last scenario, we consider the effects of eliminating the imbalance ratio of the original training datapoints via synthetic data from generative models. We follow the same setup with the augmentation use case, developing a TA-TR experiment for evalution. However here, we restrict our analysis on the conditional DGMs that have been trained to model $p(x|y)$, and *extract synthetic samples corresponding to the minority category* by setting y to the minority category. We also sample minority datapoints from GReaT starting the autoregressive generation via setting the feature y to the value of the minority class. Hence, we can restore the balance among the categories of each dataset. Subsequently, we provide the downstream model with the balanced version of the data as the new richer training dataset. As observed in Table 2 the generative models are capable to only marginally outperform the original data and the SMOTE baseline in 2 of the 4 considered datasets.

4. Conclusions

We investigated the effects of generative model-based data augmentation strategies on downstream learning tasks for commonly used tabular benchmarks. We experimented with data imbalance and scarcity learning scenarios. We deliberately excluded privacy and fairness considerations from this paper to focus on the usefulness of ML training. We found that, despite ongoing advances in tabular generative modeling, popular state-of-the-art methods are not yet able to consistently outperform the ML efficacy of using only the original training datapoints for practical use cases, such as data summarization, augmentation, and learning in imbalanced datasets. We believe that including experiments that simulate real-world situations where practitioners might need synthetic data is often missing from the literature, and should be seen as a necessary prerequisite for improving tabular synthetic data generation methods. As future work, we will focus on integrating extensive HPO, adding downstream task-informed regularization terms over training of the generative models, as well as using pre-trained LLMs that are more relevant to the domain of the training data.

References

Ai, Q., Wang, P., He, L., Wen, L., Pan, L., and Xu, Z. Generative oversampling for imbalanced data via majority-guided vae. In *International Conference on Artificial Intelligence and Statistics*, pp. 3315–3330. PMLR, 2023.

Borisov, V., Sessler, K. S., Leemann, T., Pawelczyk, M., and Kasneci, G. Language models are realistic tabular data generators. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Dua, D., Graff, C., et al. UCI machine learning repository. 2017.

Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. Neural spline flows. In *Advances in Neural Information Processing Systems*, 2019.

Elor, Y. and Averbuch-Elor, H. To SMOTE, or not to SMOTE? *CoRR*, abs/2201.08528, 2022. URL https://arxiv.org/abs/2201.08528.

Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M., and Smola, A. AutoGluon-Tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

Kim, J., Lee, C., Shin, Y., Park, S., Kim, M., Park, N., and Cho, J. SOS: Score-based oversampling for tabular data. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pp. 762–772, 2022.

Kim, J., Lee, C., and Park, N. STaSy: Score-based tabular data synthesis. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

Kohavi, R. et al. Scaling up the accuracy of naive-Bayes classifiers: A decision-tree hybrid. In *KDD*, volume 96, pp. 202–207, 1996.

Kotelnikov, A., Baranchuk, D., Rubachev, I., and Babenko, A. TabDDPM: Modelling tabular data with diffusion models. *arXiv preprint arXiv:2209.15421*, 2022.

Lemaître, G., Nogueira, F., and Aridas, C. K. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017. URL http://jmlr.org/papers/v18/16-365.

Liu, T., Qian, Z., Berrevoets, J., and van der Schaar, M. GOGGLE: Generative modelling for tabular data by learning relational structure. In *International Conference on Learning Representations*, 2023.

Mirza, M. and Osindero, S. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(1), jan 2021. ISSN 1532-4435.

Qian, Z., Cebere, B.-C., and van der Schaar, M. Synthcity: facilitating innovative use cases of synthetic data in different data modalities, 2023. URL https://arxiv.org/abs/2301.07573.

Quinlan, J. R., Compton, P. J., Horn, K., and Lazarus, L. Inductive knowledge acquisition: a case study. In *Proceedings of the Second Australian Conference on Applications of expert systems*, pp. 137–156, 1987.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *NeurIPS Energy Efficient Machine Learning and Cognitive Computing Workshop*, 2019.

van Breugel, B., Kyono, T., Berrevoets, J., and van der Schaar, M. DECAF: Generating fair synthetic data using causally-aware generative networks. In *Advances in Neural Information Processing Systems*, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Vietri, G., Archambeau, C., Aydore, S., Brown, W., Kearns, M., Roth, A., Siva, A., Tang, S., and Wu, S. Z. Private synthetic data for multitask learning and marginal queries. *Advances in Neural Information Processing Systems*, 2022.

Xu, L., Skoularidou, M., Cuesta-Infante, A., and Veeramachaneni, K. Modeling tabular data using conditional gan. In *Advances in Neural Information Processing Systems*, 2019.

Table 3. Feature names of the datasets.

adult	age, workclass, fnlwgt, education, education-num, marital-status, occupation, relationship, race, sex, capital-gain, capital-loss, hours-per-week, native-country, **income**
churn	Age, Frequent Flyer, Annual Income, Class of user, Number of times Services Opted during recent years, Company account of user synced To Social Media, Whether the customer book lodgings / Hotels using company services, **Target**
sick	age, sex, on thyroxine, query on thyroxine, on antithyroid medication, sick, pregnant, thyroid surgery, I131 treatment, query hypothyroid, query hyperthyroid, lithium, goitre, tumor, hypopituitary, psych, TSH measured, TSH, T3 measured, T3, TT4 measured, TT4, T4U measured, T4U, FTI measured, FTI, TBG measured, referral source, **binaryClass**
heloc	External Risk Estimate, Months Since Oldest Trade Open, Months Since Most Recent Trade Open, Average Months in File, Number of Satisfactory Trades, Number of Trades 60+ Days Past Due - Public Records or Derogatory Public Records, Number of Trades 90+ Days Past Due - Public Records or Derogatory Public Records, Percentage of Trades Never Delinquent, Months Since Most Recent Delinquency, Maximum Delinquency 2 Public Records in Last 12 Months, Maximum Delinquency Ever, Number of Total Trades, Number of Trades Open in Last 12 Months, Percentage of Installment Trades, Months Since Most Recent Inquiry excluding 7 days, Number of Inquiries in Last 6 Months, Number of Inquiries in Last 6 Months excluding 7 days, Net Fraction Revolving Burden, Net Fraction Installment Burden, Number of Revolving Trades with Balance, Number of Installment Trades with Balance, Number of Bank/National Trades with High Utilization, Percentage of Trades with Balance, **Risk Performance**
california	MedInc, HouseAge, AveRooms, AveBedrms, Population, AveOccup, Latitude, Longitude, **target**

A. Dataset details

In this section we include some further details on the datasets that we used for our experiments. In Table 3 we outline the names of the dataset features used for our generative model training, indicating in bold the target variable for the downstream discrimative learning task. Note that, with the exception of GReaT, all generative methods are agnostic to the semantics of the features (only require the column type to apply the right encoding and preprocessing of the corresponding feature). On the other hand, for GReaT feature semantics directly interact with synthetic data quality, as the datapoints will be understood by the corresponding language model as a sentence of the form ”`feature_name_0` *is* `feature_value_0`, `feature_name_1` *is* `feature_value_1`, . . . ”, adopting the pre-trained word embeddings for the features.

In Table 4 we disclose the original sources of the datasets.

Table 4. URLs for real-world datasets of the study.

Dataset	URL
adult (Kohavi et al., 1996; Dua et al., 2017)	`https://archive.ics.uci.edu/ml/datasets/Adult/`
churn	`https://www.kaggle.com/datasets/tejashvi14/tour-travels-customer-churn-prediction`
sick (Quinlan et al., 1987; Dua et al., 2017)	`https://www.openml.org/search?type=data&sort=runs&id=38&status=active`
heloc	`https://www.kaggle.com/datasets/averkiyoliabev/home-equity-line-of-creditheloc`
california	`https://www.kaggle.com/datasets/camnugent/california-housing-prices`

B. Additional results

In Figures 7 to 12 we present further predictive performances metrics and downstream models used for the evaluation of the synthetic data utility in the usecases of our experiments.

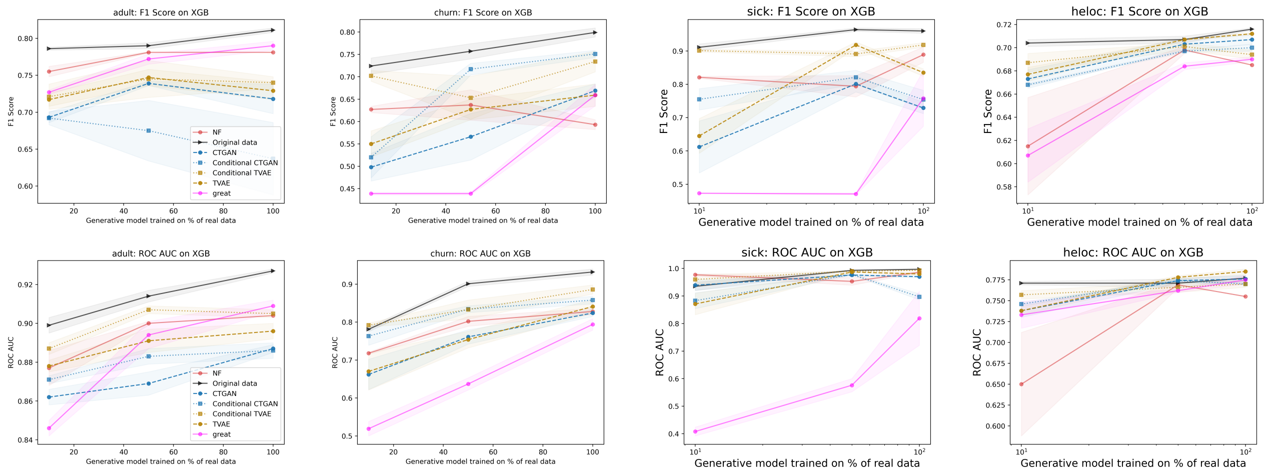


Figure 7. (*Synthetic data quality experiment*) F1-score and ROC AUC on a downstream XGB model for all classification datasets.

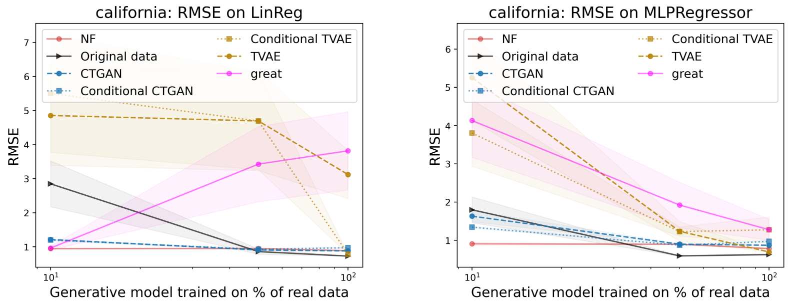


Figure 8. (*Synthetic data quality*) RMSE for downstream regression tasks on a linear model and a multi-layered perceptron.

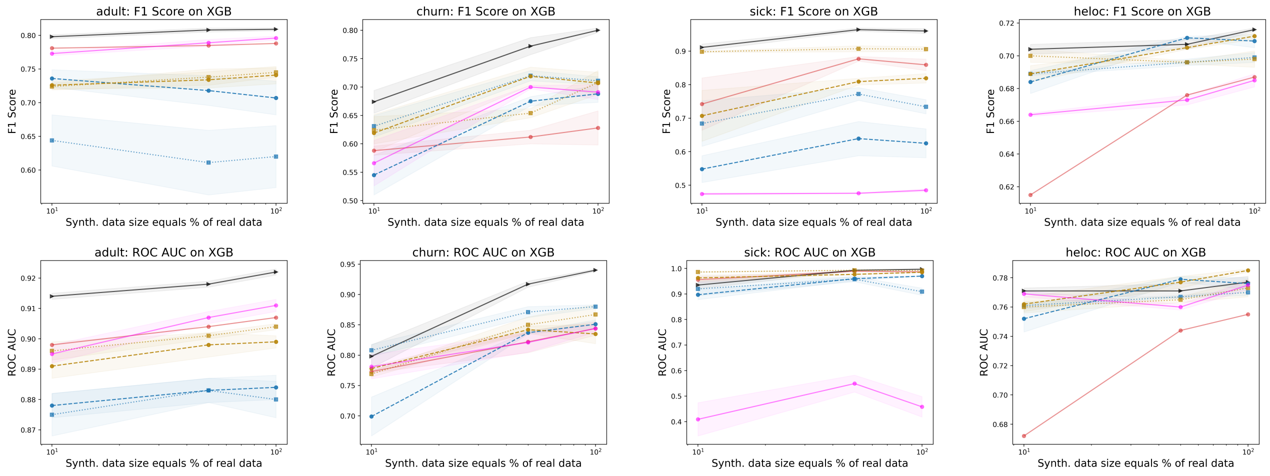


Figure 9. (*Data summarization experiment*) F1-score and ROC AUC on a downstream XGB model for all classification datasets.

C. Reproducibility

Here we add some additional details on our experimentation for reproducibility. The selected hyperparameters for the trained DGMs are provided in Table 5. Regarding the downstream classifiers, corresponding search spaces for HPO are displayed in Table 6.

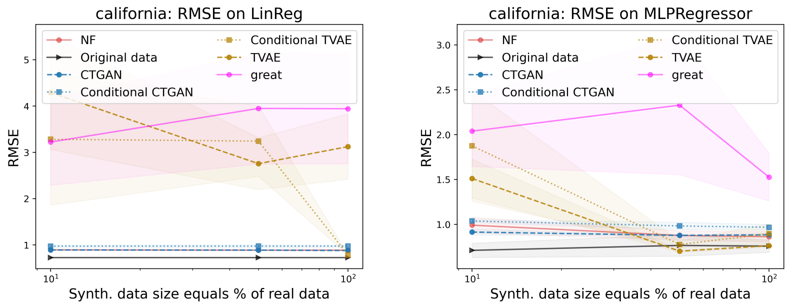


Figure 10. (*Data summarization experiment*) Predictive metrics for downstream regression tasks on a linear model and a multi-layered perceptron.

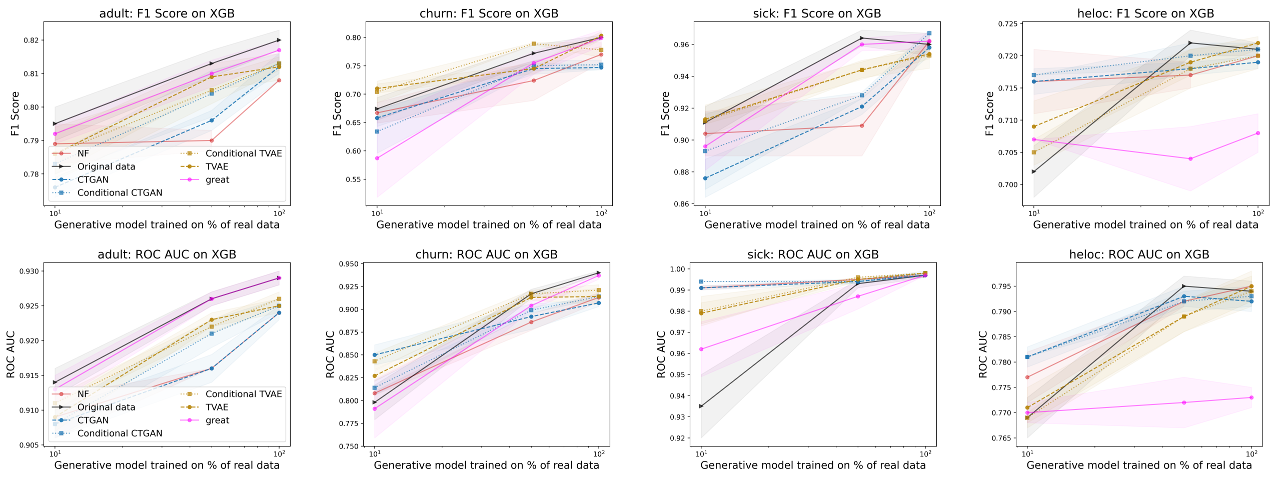


Figure 11. (*Augmentation experiment*) F1-score and ROC AUC on a downstream XGB model for all classification datasets.

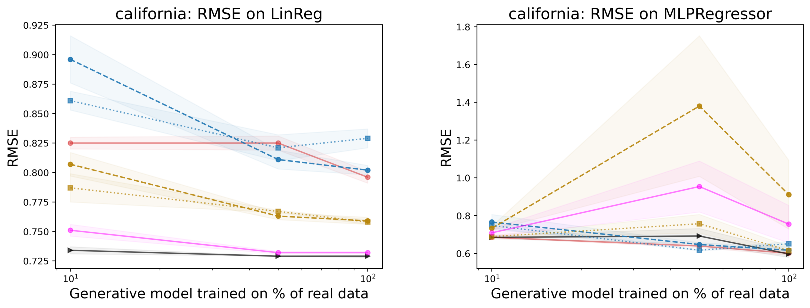


Figure 12. (*Data augmentation experiment*) Predictive metrics for downstream regression tasks on a linear model and a multi-layered perceptron.

TVAE	n_units_embedding=500,
	lr=5e-4,
	weight_decay=1e-5,
	batch_size=1000,
	decoder_n_layers_hidden=2,
	decoder_n_units_hidden=256,
	decoder_nonlin="leaky_relu",
	decoder_dropout=0.1,
	encoder_n_layers_hidden=3,
	encoder_n_units_hidden=256,
	encoder_nonlin="leaky_relu",
	encoder_dropout=0.1,
	loss_factor=1,
	data_encoder_max_clusters=10,
	clipping_value=1,
	patience=500

CTGAN	generator_n_layers_hidden=2,
	generator_n_units_hidden=256,
	generator_nonlin="relu",
	generator_dropout=0.1,
	generator_opt_betas=(0.9, 0.999),
	discriminator_n_layers_hidden=2,
	discriminator_n_units_hidden=256,
	discriminator_nonlin="leaky_relu",
	discriminator_n_iter=1,
	discriminator_dropout=0.1,
	discriminator_opt_betas=(0.9, 0.999),
	lr=5e-4,
	weight_decay=1e-3,
	batch_size=1000,
	clipping_value=1,
	lambda_gradient_penalty=10,
	encoder_max_clusters=10,
	patience=500,

NF	n_layers_hidden=2,
	n_units_hidden=256,
	batch_size=1000,
	num_transform_blocks=1,
	dropout=0.1,
	batch_norm=False,
	num_bins=8,
	tail_bound=3,
	lr=5e-4,
	apply_unconditional_transform=True,
	base_distribution="standard_normal",
	linear_transform_type="permutation",
	base_transform_type="rq-autoregressive",
	encoder_max_clusters=10,
	n_iter_min=100,
	patience=500,

Table 5. Hyperparameters used for our experiments with deep generative models.

XGB
max_depth $\in \{2, 4, 8, 12\}$,
learning_rate $\in \{1e-6, 1e-4, 1e-3, 1e-2, 1e-1, 1.0\}$,
$\alpha \in \{0.0, 1e-6, 1e-4, 1e-2, 1e-1, 1, 1.0\}$,
$\lambda \in \{1e-8, 1e-6, 1e-4, 1e-2, 1e-1, 1, 1.0\}$,
$\gamma \in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 1.0\}$,
min_child_weight $\in \{1, 3, 5, 7, 9\}$
MLP Regressor
hidden_layer_sizes $\in \{(100,), (200,)\}$,
$\alpha \in \{1e-4, 1e-3\}$,
learning_rate $\in \{$"constant", "invscaling"$\}$,
learning_rate_init $\in \{1e-3, 1e-4\}$

Table 6. Search spaces for hyperparameter optimization on downstream models.