# TabDDPM: Modelling Tabular Data with Diffusion Models

**Akim Kotelnikov**
HSE, Yandex
ya@akotelnikov.ru

**Dmitry Baranchuk**
Yandex

**Ivan Rubachev**
HSE, Yandex

**Artem Babenko**
Yandex

## Abstract

Denoising diffusion probabilistic models are currently becoming the leading paradigm of generative modeling for many important data modalities. Being the most prevalent in the computer vision community, diffusion models have also recently gained some attention in other domains, including speech, NLP, and graph-like data. In this work, we investigate if the framework of diffusion models can be advantageous for general tabular problems, where datapoints are typically represented by vectors of heterogeneous features. The inherent heterogeneity of tabular data makes it quite challenging for accurate modeling, since the individual features can be of completely different nature, i.e., some of them can be continuous and some of them can be discrete. To address such data types, we introduce TabDDPM — a diffusion model that can be universally applied to any tabular dataset and handles any type of feature. We extensively evaluate TabDDPM on a wide set of benchmarks and demonstrate its superiority over existing GAN/VAE alternatives, which is consistent with the advantage of diffusion models in other fields. Additionally, we show that TabDDPM is eligible for privacy-oriented setups, where the original datapoints cannot be publicly shared. The source code of TabDDPM and our experiments is available at `https://github.com/rotot0/tab-ddpm`.

## 1 Introduction

Denoising diffusion probabilistic models (DDPM) (Sohl-Dickstein et al., 2015; Ho et al., 2020) have recently become an object of great research interest in the generative modelling community since they often outperform the alternative approaches both in terms of the realism of individual samples and their diversity (Dhariwal & Nichol, 2021). The most impressive successes of DDPM were demonstrated in the domain of natural images (Dhariwal & Nichol, 2021; Saharia et al., 2022; Rombach et al., 2022), where the advantages of diffusion models are successfully exploited in applications, such as colorization (Song et al., 2021), inpainting (Song et al., 2021), segmentation Baranchuk et al. (2021), super-resolution (Saharia et al., 2021; Li et al., 2021), semantic editing (Meng et al., 2021) and others. Beyond computer vision, the DDPM framework is also investigated in other fields, such as NLP (Austin et al., 2021; Li et al., 2022), waveform signal processing (Kong et al., 2020; Chen et al., 2020), molecular graphs (Jing et al., 2022; Hoogeboom et al., 2022), time series (Tashiro et al., 2021), testifying the universality of diffusion models across a wide range of problems.

The aim of our work is to understand if the universality of DDPM can be extended to the case of general tabular problems, which are ubiquitous in various industrial applications that include data described by a set of heterogeneous features. For many such applications, the demand for high-quality generative models is especially acute because of the modern privacy regulations, like GDPR, which prevent publishing real user data, while the synthetic data produced by generative models can be shared. Training a high-quality model of tabular data, however, can be more challenging compared to computer vision or NLP due to the heterogeneity of individual features and relatively small sizes of typical tabular datasets. In our paper, we show that despite these two intricacies, the diffusion models can successfully approximate typical distributions of tabular data, leading to state-of-the-art performance on most of the benchmarks.

In more detail, the main contributions of our work are the following:

1. We introduce TabDDPM — the simplest design of DDPM for tabular problems that can be applied to any tabular task and can work with mixed data, which includes both numerical and categorical features.
2. We demonstrate that TabDDPM outperforms the alternative approaches designed for tabular data, including GAN-based and VAE-based models from the literature, and illustrate the sources of this advantage for several datasets.
3. We show that data produced by TabDDPM appears to be a "sweet spot" for privacy-concerned scenarios when synthetics are used to substitute the real user data that cannot be shared.

The source code of TabDDPM is publicly available [1].

## 2 Related Work

**Diffusion models** (Sohl-Dickstein et al., 2015; Ho et al., 2020) are a paradigm of generative modelling that aims to approximate the target distribution by the endpoint of the Markov chain, which starts from a given parametric distribution, typically a standard Gaussian. Each Markov step is performed by a deep neural network that effectively learns to invert the diffusion process with a known Gaussian kernel. Ho et al. demonstrated the equivalence of diffusion models and score matching (Song & Ermon, 2019; 2020), showing them to be two different perspectives on the gradual conversion of a simple known distribution into a target distribution via the iterative denoising process. Several recent works (Nichol, 2021; Dhariwal & Nichol, 2021) have developed more powerful model architectures as well as different advanced learning protocols, which led to the "victory" of DDPM over GANs in terms of generative quality and diversity in the computer vision field. In our work, we demonstrate that one can also successfully use diffusion models for tabular problems.

**Generative models for tabular problems** are currently an active research direction in the machine learning community (Xu et al., 2019; Engelmann & Lessmann, 2021; Jordon et al., 2018; Fan et al., 2020; Torfi et al., 2022; Zhao et al., 2021; Kim et al., 2021; Zhang et al., 2021; Nock & Guillame-Bert, 2022; Wen et al., 2022) since high-quality synthetic data is of large demand for many tabular tasks. First, the tabular datasets are often limited in size, unlike in vision or NLP problems, for which huge "extra" data is available on the Internet. Second, proper synthetic datasets do not contain actual user data, therefore they are not subject to GDPR-like regulations and can be publicly shared without violation of anonymity. The recent works have developed a large number of models, including tabular VAEs (Xu et al., 2019) and GAN-based approaches (Xu et al., 2019; Engelmann & Lessmann, 2021; Jordon et al., 2018; Fan et al., 2020; Torfi et al., 2022; Zhao et al., 2021; Kim et al., 2021; Zhang et al., 2021; Nock & Guillame-Bert, 2022; Wen et al., 2022). By extensive evaluations on a large number of public benchmarks, we show that our TabDDPM model surpasses the existing alternatives, often by a large margin.

**"Shallow" synthetics generation.** Unlike unstructured images or natural texts, tabular data is typically structured, i.e., the individual features are often interpretable and it is not clear if their modelling requires several layers of "deep" architectures. Therefore, the simple interpolation techniques, like SMOTE (Chawla et al., 2002) (originally proposed to address class-imbalance) can serve as simple and powerful solutions as demonstrated in (Camino et al., 2020), where SMOTE is shown to outperform tabular GANs for minor class oversampling. In the experiments, we demonstrate the advantage of synthetics produced by TabDDPM over synthetics produced by interpolation techniques from the privacy-preserving perspective.

## 3 Background

**Diffusion models** (Sohl-Dickstein et al., 2015; Ho et al., 2020) are likelihood-based generative models that handle the data through forward and reverse Markov processes. The forward process

[1] https://github.com/rotot0/tab-ddpm

$q\left(x_{1:T}|x_0\right)=\prod_{t=1}^{T} q\left(x_t|x_{t-1}\right)$ gradually adds noise to an initial sample $x_0$ from the data distribution $q\left(x_0\right)$ sampling noise from the predefined distributions $q\left(x_t|x_{t-1}\right)$ with variances $\{\beta_1, ..., \beta_T\}$.

The reverse diffusion process $p\left(x_{0:T}\right)=\prod_{t=1}^{T} p\left(x_{t-1}|x_t\right)$ gradually denoises a latent variable $x_T \sim q\left(x_T\right)$ and allows generating new data samples from $q(x_0)$. Distributions $p\left(x_{t-1}|x_t\right)$ are usually unknown and approximated by a neural network with parameters $\theta$. These parameters are learned from the data by optimizing a variational lower bound:

$$\log q\left(x_0\right) \geq \mathbb{E}_{q(x_0)}\big[\underbrace{\log p_\theta\left(x_0|x_1\right)}_{L_0} - \underbrace{KL\left(q\left(x_T|x_0\right)|q\left(x_T\right)\right)}_{L_T} - \sum_{t=2}^{T} \underbrace{KL\left(q\left(x_{t-1}|x_t, x_0\right)|p_\theta\left(x_{t-1}|x_t\right)\right)}_{L_t}\big] \tag{1}$$

**Gaussian diffusion models** operate in continuous spaces $(x_t \in \mathbb{R}^n)$ where forward and reverse processes are characterized by Gaussian distributions:

$$\begin{aligned}
&q\left(x_t|x_{t-1}\right) := \mathcal{N}\left(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t I\right)\\
&q\left(x_T\right) := \mathcal{N}\left(x_T; 0, I\right)\\
&p_\theta\left(x_{t-1}|x_t\right) := \mathcal{N}\left(x_{t-1}; \mu_\theta\left(x_t, t\right), \Sigma_\theta\left(x_t, t\right)\right)
\end{aligned}$$

Ho et al. (2020) suggest using diagonal $\Sigma_\theta\left(x_t, t\right)$ with a constant $\sigma_t$ and computing $\mu_\theta\left(x_t, t\right)$ as a function of $x_t$ and $\epsilon_\theta(x_t, t)$:

$$\mu_\theta\left(x_t, t\right) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta\left(x_t, t\right)\right)$$

where $\alpha_t := 1 - \beta_t,\ \bar{\alpha}_t := \prod_{i \leq t} \alpha_i$ and $\epsilon_\theta(x_t, t)$ predicts a "groundtruth" noise component $\epsilon$ for the noisy data sample $x_t$. In practice, the objective (1) can be simplified to the sum of mean-squared errors between $\epsilon_\theta(x_t, t)$ and $\epsilon$ over all timesteps $t$:

$$L_t^{simple} = \mathbb{E}_{x_0, \epsilon, t} \|\epsilon - \epsilon_\theta(x_t, t)\|_2^2 \tag{2}$$

**Multinomial diffusion models** (Hoogeboom et al., 2021) are designed to generate categorical data where $x_t \in \{0, 1\}^K$ is a one-hot encoded categorical variable with $K$ values. The multinomial forward diffusion process defines $q\left(x_t|x_{t-1}\right)$ as a categorical distribution that corrupts the data by uniform noise over $K$ classes:

$$\begin{aligned}
&q(x_t|x_{t-1}) := Cat\left(x_t; \left(1-\beta_t\right) x_{t-1} + \beta_t/K\right)\\
&q\left(x_T\right) := Cat\left(x_T; 1/K\right)\\
&q\left(x_t|x_0\right) = Cat\left(x_t; \bar{\alpha}_t x_0 + \left(1-\bar{\alpha}_t\right)/K\right)
\end{aligned}$$

From the equations above, the posterior $q(x_{t-1}|x_t, x_0)$ can be derived:

$$q\left(x_{t-1}|x_t, x_0\right) = Cat\left(x_{t-1}; \pi / \sum_{k=1}^{K} \pi_k\right)$$

where $\pi = [\alpha_t x_t + (1-\alpha_t)/K] \odot [\bar{\alpha}_{t-1} x_0 + (1-\bar{\alpha}_{t-1})/K]$.

The reverse distribution $p_\theta\left(x_{t-1}|x_t\right)$ is parameterized as $q\left(x_{t-1}|x_t, \hat{x}_0(x_t, t)\right)$, where $\hat{x}_0$ is predicted by a neural network. Then, the model is trained to maximize the variational lower bound (1).

## 4 TabDDPM

In this section, we describe the design of TabDDPM as well as its main hyperparameters, which affect the model's effectiveness.

**TabDDPM** uses the multinomial diffusion to model the categorical and binary features, and the Gaussian diffusion to model the numerical ones. In more detail, for a tabular data sample $x = [x_{num}, x_{cat_1}, ..., x_{cat_C}]$, that consists of $N_{num}$ numerical features $x_{num} \in \mathbb{R}^{N_{num}}$ and $C$

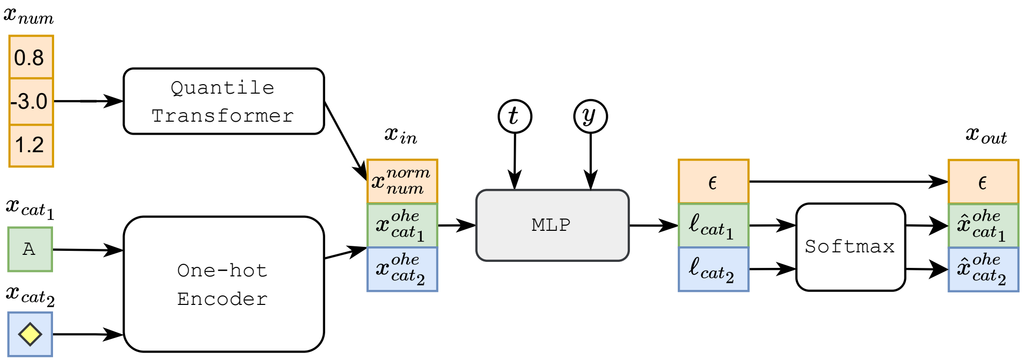

Figure 1: TabDDPM scheme for classification problems; $t$, $y$ and $\ell$ denote a diffusion timestep, a class label, and logits, respectively.

categorical features $x_{cat_i}$ with $K_i$ categories each, our model takes one-hot encoded versions of categorical features as an input (i.e. $x_{cat_i}^{ohe} \in \{0,1\}^{K_i}$) and normalized numerical features. Therefore, the input $x_0$ has a dimensionality of $(N_{num} + \sum K_i)$. For preprocessing, we use the gaussian quantile transformation from the scikit-learn library (Pedregosa et al., 2011). Each categorical feature is handled by a separate forward diffusion process, i.e., the noise components for all features are sampled independently. The reverse diffusion step in TabDDPM is modelled by a multi-layer neural network that has an output of the same dimensionality as $x_0$, where the first $N_{num}$ coordinates are the predictions of $\epsilon$ for the Gaussian diffusion and the rest are the predictions of $x_{cat_i}^{ohe}$ for the multinomial diffusions.

The TabDDPM model for the classification problems is schematically presented on Figure 1. The model is trained by minimizing a sum of mean-squared error $L_t^{simple}$ (Equation (2)) for the Gaussian diffusion term and the KL divergences $L_t^i$ for each multinomial diffusion term (Equation (1)). The total loss of multinomial diffusions is additionally divided by the number of categorical features.

$$L_t^{TabDDPM} = L_t^{simple} + \frac{\sum_{i \le C} L_t^i}{C} \qquad (3)$$

For classification datasets, we use a class-conditional model, i.e. $p_\theta(x_{t-1}|x_t, y)$ is learned. For regression datasets, we consider a target value as an additional numerical feature, and the joint distribution is learned.

| Hyperparameter | Search space |
|---|---|
| Learning rate | $\mathrm{LogUniform}[0.00001, 0.003]$ |
| Batch size | $\mathrm{Cat}\{256, 4096\}$ |
| Diffusion timesteps | $\mathrm{Cat}\{100, 1000\}$ |
| Training iterations | $\mathrm{Cat}\{5000, 10000, 20000\}$ |
| # MLP layers | $\mathrm{Int}\{2, 4, 6, 8\}$ |
| MLP width of layers | $\mathrm{Int}\{128, 256, 512, 1024\}$ |
| Proportion of samples | $\mathrm{Float}\{0.25, 0.5, 1, 2, 4, 8\}$ |
| Dropout | 0.0 |
| Scheduler | cosine (Nichol, 2021) |
| Gaussian diffusion loss | MSE |
| Number of tuning trials | 50 |

Table 1: The main hyperparameters of TabDDPM.

To model the reverse process, we use a simple MLP architecture adapted from (Gorishniy et al., 2021):

$$\begin{aligned} \texttt{MLP}(x) &= \texttt{Linear}\left(\texttt{MLPBlock}\left(\ldots\left(\texttt{MLPBlock}(x)\right)\right)\right) \\ \texttt{MLPBlock}(x) &= \texttt{Dropout}(\texttt{ReLU}(\texttt{Linear}(x))) \end{aligned} \qquad (4)$$

As in (Nichol, 2021; Dhariwal & Nichol, 2021), a tabular input $x_{in}$, a timestep $t$, and a class label $y$ are processed as follows.

$$\begin{aligned} t\_emb &= \texttt{Linear}(\texttt{SiLU}(\texttt{Linear}(\texttt{SinTimeEmb}(t)))) \\ y\_emb &= \texttt{Embedding}(y) \\ x &= \texttt{Linear}(x_{in}) + t\_emb + y\_emb \end{aligned} \qquad (5)$$

| Abbr | Name | # Train | # Validation | # Test | # Num | # Cat | Task type |
|---|---|---|---|---|---|---|---|
| AB | Abalone | 2672 | 669 | 836 | 7 | 1 | Regression |
| AD | Adult ROC | 26048 | 6513 | 16281 | 6 | 8 | Binclass |
| BU | Buddy | 12053 | 3014 | 3767 | 4 | 5 | Multiclass |
| CA | California Housing | 13209 | 3303 | 4128 | 8 | 0 | Regression |
| CAR | Cardio | 44800 | 11200 | 14000 | 5 | 6 | Binclass |
| CH | Churn Modelling | 6400 | 1600 | 2000 | 7 | 4 | Binclass |
| DI | Diabetes | 491 | 123 | 154 | 8 | 0 | Binclass |
| FB | Facebook Comments Volume | 157638 | 19722 | 19720 | 50 | 1 | Regression |
| GE | Gesture Phase | 6318 | 1580 | 1975 | 32 | 0 | Multiclass |
| HI | Higgs Small | 62751 | 15688 | 19610 | 28 | 0 | Binclass |
| HO | House 16H | 14581 | 3646 | 4557 | 16 | 0 | Regression |
| IN | Insurance | 856 | 214 | 268 | 3 | 3 | Regression |
| KI | King | 13832 | 3458 | 4323 | 17 | 3 | Regression |
| MI | MiniBooNE | 83240 | 20811 | 26013 | 50 | 0 | Binclass |
| WI | Wilt | 3096 | 775 | 968 | 5 | 0 | Binclass |

Table 2: List of datasets used for the evaluation and their descriptions.

where `SinTimeEmb` refers to a sinusoidal time embedding as in (Nichol, 2021; Dhariwal & Nichol, 2021) with a dimension of 128. All `Linear` layers in Equation 5 have a fixed projection dimension 128.

**Hyperparameters** in TabDDPM are essential since in the experiments we observed them having a strong influence on the model effectiveness. Table 1 lists the main hyperparameters as well as the search spaces for each of them, which we recommend to use. The process of tuning is described in detail in the experimental section.

## 5 Experiments

In this section, we extensively evaluate TabDDPM against existing alternatives.

**Datasets.** For systematic investigation of the performance of tabular generative models, we consider a diverse set of 15 real-world public datasets. These datasets have various sizes, nature, number of features, and their distributions. Most datasets were previously used for tabular model evaluation in (Zhao et al., 2021; Gorishniy et al., 2021). The full list of datasets and their properties are presented in Table 2.

**Baselines.** Since the number of generative models proposed for tabular data is enormous, we evaluate TabDDPM only against the leading approaches from each paradigm of generative modelling. Also, we consider only the baselines with the published source code.

- **TVAE** (Xu et al., 2019) — the state-of-the-art variational auto-encoder for tabular data generation. To the best of our knowledge, there are no alternative VAE-like models that outperform TVAE and have public source code.
- **CTABGAN** (Zhao et al., 2021) — a recent GAN-based model that is shown to outperform the existing tabular GANs on a diverse set of benchmarks. This approach cannot handle regression tasks.
- **CTABGAN+**(Zhao et al., 2022) — an extension of the **CTABGAN** model that was published in the very recent preprint. We are not aware of the GAN-based model for tabular data that is proposed after **CTABGAN+** and has a public source code.
- **SMOTE** (Chawla et al., 2002) — a “shallow” interpolation-based method that ”generates” a synthetic point as a convex combination of a real data point and its $k$-th nearest neighbor from the dataset. This method was originally proposed for minor class oversampling. Here, we generalize it and apply it to synthetic data generation as a simple sanity check.

**Evaluation measure.** Our primary evaluation measure is *machine learning (ML) efficiency* (or utility) (Xu et al., 2019). In more detail, ML efficiency quantifies the performance of classification or regression models that are trained on synthetic data and evaluated on the real test set. Intuitively,

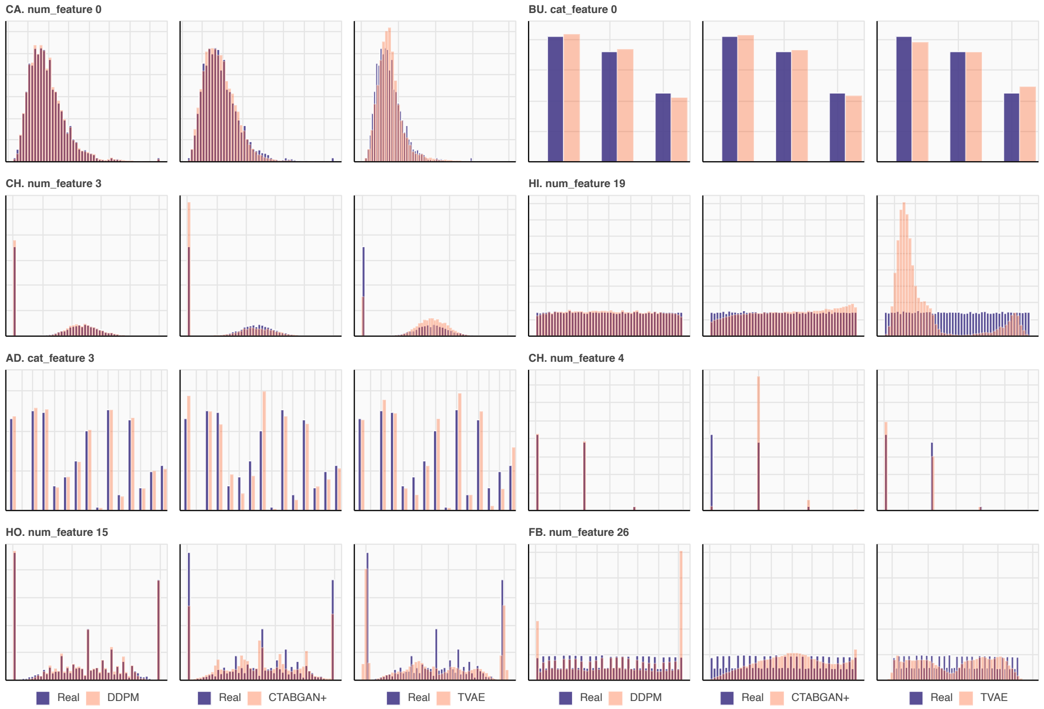

Figure 2: The individual feature distributions for the real data and the data generated by TabDDPM, CTABGAN+, and TVAE. TabDDPM produces more realistic feature distributions than alternatives in most cases.

models trained on high-quality synthetics should be competitive (or even superior) to models trained on real data. In our experiments, we use two evaluation protocols to compute ML efficiency. In the first protocol, which is more common in the literature (Xu et al., 2019; Zhao et al., 2021; Kim et al., 2022), we compute an average efficiency with respect to a set of diverse ML models (logistic regression, decision tree, and others). In the second protocol, we evaluate ML efficiency only with respect to the CatBoost model (Prokhorenkova et al., 2018), which is the leading GBDT implementation providing state-of-the-art performance on tabular tasks. In our experiments in subsection 5.2, we show that it is crucial to use the second protocol, while the first one can often be misleading.

**Tuning process.** To tune the hyperparameters of TabDDPM and the baselines, we use the Optuna library (Akiba et al., 2019). The tuning process is guided by the values of the ML efficiency (with respect to Catboost) of the generated synthetic data on a hold-out validation dataset (the score is averaged over five different sampling seeds). The search spaces for all hyperparameters of TabDDPM are reported in Table 1 (for baselines — in Appendix B). Additionally, we demonstrate that tuning the hyperparameters using the CatBoost guidance does not introduce any sort of “Catboost-biasedness”, and the Catboost-tuned TabDDPM produces synthetics that are also superior for other models, like MLP. These results are reported in Appendix A.

### 5.1 Qualitative comparison

Here, we qualitatively investigate the ability of TabDDPM to model the individual and joint feature distributions compared to the TVAE and CTABGAN+ baselines. In particular, for each dataset, we sample a synthetic dataset from TabDDPM, TVAE, and CTABGAN+ of the same size as a real train set in a particular dataset. For classification datasets, each class is sampled according to its proportion in the real dataset. Then, we visualize the typical individual feature distributions for real and synthetic data in Figure 2. For completeness, the features of different types and distributions are presented. In most cases, TabDDPM produces more realistic feature distributions compared to TVAE and CTABGAN+. The advantage is more pronounced (1) for numerical features, which are uniformly distributed, (2) for categorical features with high cardinality, and (3) for mixed type features

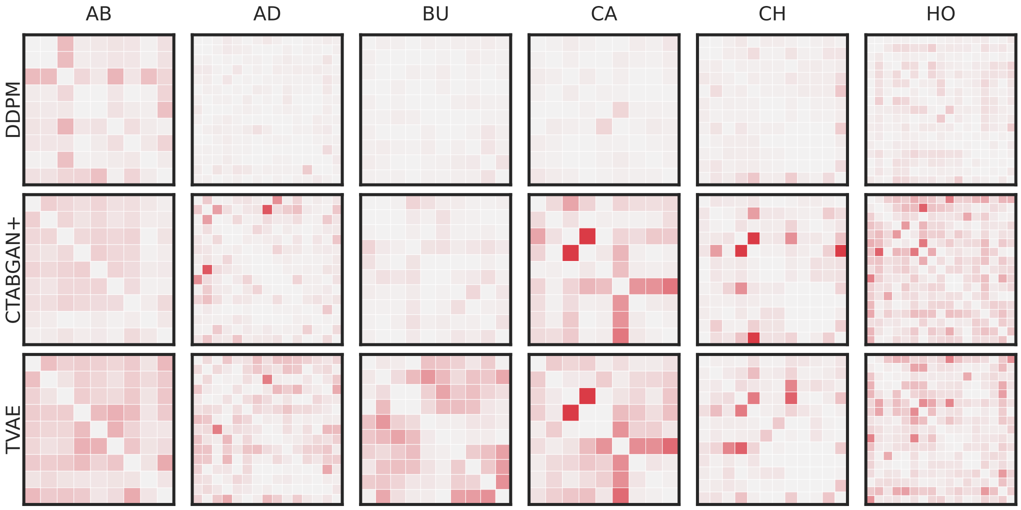

Figure 3: The absolute difference between correlation matrices computed on real and synthetic datasets. A more intensive red colour indicates a higher difference between the real and synthetic correlation values.

that combine continuous and discrete distributions. Then, we also visualize the differences between the correlation matrices computed on real and synthetic data for different datasets, see Figure 3. To compute the correlation matrices, we use the Pearson correlation coefficient for numerical-numerical correlations, the correlation Ratio for categorical-numerical cases, and Theil's U statistic between categorical features. Compared to CTABGAN+ and TVAE, TabDDPM generates synthetic datasets with more realistic pairwise correlations. These illustrations indicate that our TabDDPM model is more flexible compared to alternatives and produces superior synthetic data.

### 5.2 Machine Learning efficiency

In this section, we compare TabDDPM to alternative generative models in terms of machine learning efficiency. From each generative model, we sample a synthetic dataset with the size of a real train set in proportion from Table 1. This synthetic data is then used to train a classification/regression model, which is then evaluated using the real test set. In our experiments, classification performance is evaluated by the F1 score, and regression performance is evaluated by the R2 score. We use two protocols:

1. First, we compute average ML efficiency for a diverse set of ML models, as performed in previous works (Xu et al., 2019; Zhao et al., 2021; Kim et al., 2022). This set includes Decision Tree, Random Forest, Logistic Regression (or Ridge Regression) and MLP models from the scikit-learn library (Pedregosa et al., 2011) with the default hyperparameters except for "max-depth" equals 28 for Decision Tree and Random Forest, "maximum iterations" equals 500 for Logistic and Ridge regressions, and "maximum iterations" equals 100 for MLPs.
2. Second, we compute ML efficiency with respect to the current state-of-the-art model for tabular data. Specifically, we consider CatBoost (Prokhorenkova et al., 2018) and MLP architecture from (Gorishniy et al., 2021) for evaluation. CatBoost and MLP hyperparameters are thoroughly tuned on each dataset using the search spaces from (Gorishniy et al., 2021). We argue that this evaluation protocol demonstrates the practical value of synthetic data more reliably since in most real scenarios practitioners are not interested in using weak and suboptimal classifiers/regressors.

**Main results.** The ML efficiency values computed by both protocols are presented in Table 3 and in Table 4. The ML efficiency for the tuned MLP is reported in Appendix A. To compute each value, we average the results over five random seeds for synthetics generation, and for each generated

|  | AB ($R2$) | AD ($F1$) | BU ($F1$) | CA ($R2$) | CAR ($F1$) | CH ($F1$) | DE ($F1$) | DI ($F1$) |
|---|---|---|---|---|---|---|---|---|
| TVAE | 0.238±.012 | 0.742±.001 | 0.779±.004 | −13.0±1.51 | 0.693±.002 | 0.684±.003 | 0.643±.003 | 0.712±.010 |
| CTABGAN | – | 0.737±.007 | 0.786±.008 | – | 0.684±.003 | 0.636±.010 | 0.614±.007 | 0.655±.015 |
| CTABGAN+ | 0.316±.024 | 0.730±.007 | 0.837±.006 | −7.59±.645 | **0.708±.002** | 0.650±.008 | 0.648±.008 | **0.727±.023** |
| SMOTE | **0.400±.009** | 0.750±.004 | 0.842±.003 | 0.667±.006 | 0.693±.001 | 0.690±.003 | 0.649±.003 | 0.677±.013 |
| TabDDPM | **0.392±.009** | **0.758±.005** | **0.851±.003** | **0.695±.002** | 0.696±.001 | **0.693±.003** | **0.659±.003** | 0.675±.011 |
| Real | 0.423±.009 | 0.750±.006 | 0.845±.004 | 0.663±.002 | 0.683±.002 | 0.679±.003 | 0.648±.003 | 0.699±.012 |

|  | FB ($R2$) | GE ($F1$) | HI ($F1$) | HO ($R2$) | IN ($R2$) | KI ($R2$) | MI ($F1$) | WI ($F1$) |
|---|---|---|---|---|---|---|---|---|
| TVAE | $\ll 0$ | 0.372±.006 | 0.590±.004 | 0.174±.012 | 0.470±.024 | 0.666±.006 | **0.880±.002** | 0.497±.001 |
| CTABGAN | – | 0.339±.009 | 0.539±.006 | – | – | – | 0.856±.003 | 0.656±.011 |
| CTABGAN+ | $\ll 0$ | 0.373±.009 | 0.598±.004 | 0.222±.042 | 0.669±.018 | 0.197±.051 | 0.867±.002 | 0.653±.027 |
| SMOTE | **0.651±.002** | **0.478±.005** | 0.664±.003 | 0.394±.006 | 0.709±.008 | **0.751±.005** | 0.860±.001 | **0.793±.004** |
| TabDDPM | 0.527±.005 | 0.462±.005 | **0.670±.002** | **0.426±.007** | **0.734±.007** | 0.611±.013 | 0.850±.004 | **0.792±.004** |
| Real | 0.645±.005 | 0.431±.005 | 0.663±.002 | 0.415±.007 | 0.708±.007 | 0.768±.013 | 0.850±.004 | 0.684±.004 |

Table 3: The values of machine learning efficiency computed with regards to five weak classification/regression models. Negative scores denote negative R2, which means that performance is worse than an optimal constant prediction.

|  | AB ($R2$) | AD ($F1$) | BU ($F1$) | CA ($R2$) | CAR ($F1$) | CH ($F1$) | DE ($F1$) | DI ($F1$) |
|---|---|---|---|---|---|---|---|---|
| TVAE | 0.433±.008 | 0.781±.002 | 0.864±.005 | 0.752±.001 | 0.717±.001 | 0.732±.006 | 0.656±.007 | **0.714±.039** |
| CTABGAN | – | 0.783±.002 | 0.855±.005 | – | 0.717±.001 | 0.688±.006 | 0.644±.011 | **0.731±.022** |
| CTABGAN+ | 0.467±.004 | 0.772±.003 | 0.884±.005 | 0.525±.004 | 0.733±.001 | 0.702±.012 | 0.686±.004 | **0.734±.020** |
| SMOTE | **0.549±.005** | 0.791±.002 | 0.891±.003 | **0.840±.001** | 0.732±.001 | 0.743±.005 | **0.693±.003** | 0.683±.037 |
| TabDDPM | **0.550±.010** | **0.795±.001** | **0.906±.003** | 0.836±.002 | **0.737±.001** | **0.755±.006** | **0.691±.004** | **0.740±.020** |
| Real | 0.556±.004 | 0.815±.002 | 0.906±.002 | 0.857±.001 | 0.738±.001 | 0.740±.009 | 0.688±.003 | 0.785±.013 |

|  | FB ($R2$) | GE ($F1$) | HI ($F1$) | HO ($R2$) | IN ($R2$) | KI ($R2$) | MI ($F1$) | WI ($F1$) |
|---|---|---|---|---|---|---|---|---|
| TVAE | 0.685±.003 | 0.434±.006 | 0.638±.003 | 0.493±.006 | 0.784±.010 | 0.824±.003 | 0.912±.001 | 0.501±.012 |
| CTABGAN | – | 0.392±.006 | 0.575±.004 | – | – | – | 0.889±.002 | **0.906±.019** |
| CTABGAN+ | 0.509±.011 | 0.406±.009 | 0.664±.002 | 0.504±.005 | 0.797±.005 | 0.444±.014 | 0.892±.002 | 0.798±.021 |
| SMOTE | **0.803±.002** | **0.658±.007** | **0.722±.001** | 0.662±.004 | **0.812±.002** | **0.842±.004** | 0.932±.001 | **0.913±.007** |
| TabDDPM | 0.713±.002 | 0.597±.006 | **0.722±.001** | **0.677±.010** | 0.809±.002 | **0.833±.014** | **0.936±.001** | **0.904±.009** |
| Real | 0.837±.001 | 0.636±.007 | 0.724±.001 | 0.662±.003 | 0.814±.001 | 0.907±.002 | 0.934±.000 | 0.898±.006 |

Table 4: The values of machine learning efficiency computed with regards to the state-of-the-art tuned CatBoost model.

dataset, we average over ten random seeds for training classifiers/regressors. The key observations are described below:

- In both evaluation protocols, TabDDPM significantly outperforms TVAE and CTABGAN+ on most datasets, which highlights the advantage of diffusion models for tabular data as well as demonstrated for other domains in prior works.

- The interpolation-based SMOTE method demonstrates the performance competitive to TabDDPM and often significantly outperforms the GAN/VAE approaches. Interestingly, most of the prior works on generative models for tabular data do not compare against SMOTE, while it appears to be a simple baseline, which is challenging to beat.

- While many prior works use the first evaluation protocol to compute the ML efficiency, we argue that the second one (which uses the state-of-the-art model, like CatBoost) is more appropriate. Table 3 and Table 4 show that the absolute values of classification/regression performance are much lower for the first protocol, i.e., weak classifiers/regressors are substantially inferior to CatBoost on the considered benchmarks. Therefore, one can hardly use these suboptimal models instead of CatBoost and their performance values are uninformative for practitioners. Moreover, in the first protocol, training on synthetic data is often advantageous compared to training on real data. This creates an impression that the data produced by generative models are more valuable than the real ones. However, it is not the case when one uses the tuned ML model, as in most practical scenarios. Appendix A confirms this observation for the properly tuned MLP model.

Overall, TabDDPM provides state-of-the-art generative performance and can be used as a source of high-quality synthetic data. Interestingly, in terms of ML efficiency, a simple “shallow” SMOTE method is competitive to TabDDPM, which raises the question if sophisticated deep generative models are needed. In the section below, we provide an affirmative answer to this question.

### 5.3 Privacy

Here, we demonstrate that TabDDPM is preferable to SMOTE in setups with privacy concerns, e.g., sharing the data without disclosure of personal or sensitive information. In these setups, one is interested in high-quality synthetics that do not reveal the datapoints from the original real dataset. To quantify the privacy of synthetic, we use a median Distance to Closest Record (DCR) (Zhao et al., 2021) between synthetic and real datapoints. Specifically, for each synthetic sample, we find the minimum distance to real datapoints and take the median of these distances. Low DCR values indicate that all synthetic samples are essentially copies of some real datapoints, which violates the privacy requirements. In contrast, larger DCR values indicate that the generative model can produce something “new” rather than just copies of real data. Table 5 compares the DCR values for SMOTE and TabDDPM and demonstrates the advantage of TabDDPM consistently for all datasets. We also visualize histograms of the minimal synthetic-to-real distances on Figure 4. For SMOTE, most distance values are concentrated around zero, while TabDDPM samples are better separated from real datapoints. This experiment confirms that TabDDPM synthetics while providing high ML efficiency, are also more appropriate for privacy-concerned scenarios.

| | AB | | AD | | BU | | CA | | CAR | | CH | | DE | | DI | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | score | DCR | score | DCR | score | DCR | score | DCR | score | DCR | score | DCR | score | DCR | score | DCR |
| SMOTE | 0.549 | 0.014 | 0.791 | 0.024 | 0.891 | 0.054 | 0.840 | 0.014 | 0.732 | 0.007 | 0.743 | 0.077 | 0.693 | 0.027 | 0.683 | 0.068 |
| TabDDPM | 0.550 | **0.050** | 0.795 | **0.104** | 0.906 | **0.143** | 0.836 | **0.041** | 0.737 | **0.012** | 0.755 | **0.157** | 0.691 | **0.112** | 0.740 | **0.204** |

| | FB | | GE | | HI | | HO | | IN | | KI | | MI | | WI | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | score | DCR | score | DCR | score | DCR | score | DCR | score | DCR | score | DCR | score | DCR | score | DCR |
| SMOTE | 0.803 | 0.027 | 0.658 | 0.023 | 0.722 | 0.319 | 0.662 | 0.056 | 0.812 | 0.030 | 0.842 | 0.066 | 0.932 | 0.016 | 0.913 | 0.007 |
| TabDDPM | 0.713 | **0.112** | 0.597 | **0.059** | 0.722 | **0.449** | 0.677 | **0.086** | 0.809 | **0.041** | 0.833 | **0.189** | 0.936 | **0.022** | 0.904 | **0.016** |

Table 5: ML efficiency CatBoost scores and privacy scores for SMOTE and TabDDPM models.

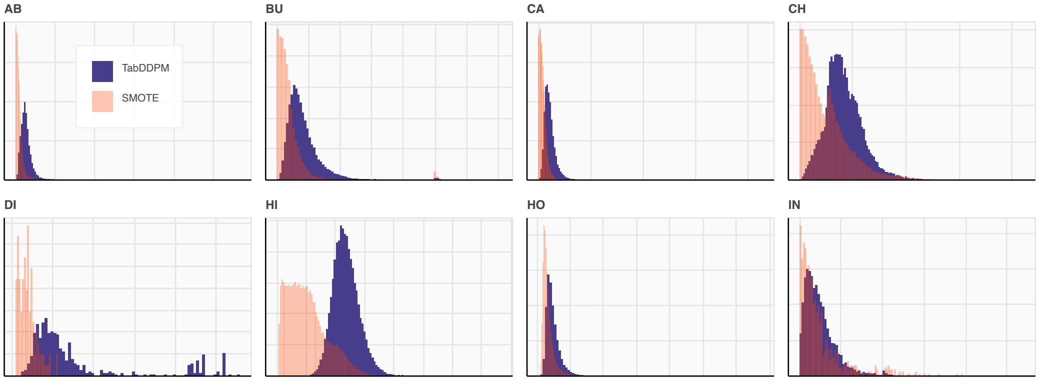

Figure 4: Histograms of minimal synthetic-to-real distances for TabDDM and SMOTE.

## 6 Conclusion

In this paper, we have investigated the prospect of the diffusion modelling framework in the field of tabular data. In particular, we have described the design of DDPM that can handle mixed data consisting of numerical, ordinal, and categorical features. We also demonstrate the importance of the model’s hyperparameters and explain the protocol of their tuning. For the most considered benchmarks, the synthetics produced by our model has consistently higher quality compared to ones produced by the GAN/VAE-based rivals and interpolation techniques, especially for the setups, where the privacy of the data must be ensured.

## REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

P. Baldi, P. Sadowski, and D. Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature Communications*, 5, 2014.

Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. *arXiv preprint arXiv:2112.03126*, 2021.

Ramiro D Camino, Christian A Hammerschmidt, et al. Oversampling tabular data with deep generative models: Is it worth the effort? 2020.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, 2020.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. 2021.

Justin Engelmann and Stefan Lessmann. Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning. *Expert Systems with Applications*, 174:114582, 2021.

Ju Fan, Tongyu Liu, Guoliang Li, Junyou Chen, Yuwei Shen, and Xiaoyong Du. Relational data synthesis using generative adversarial networks: A design space exploration. *arXiv preprint arXiv:2008.12763*, 2020.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. 2020.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34:12454–12465, 2021.

Emiel Hoogeboom, Victor Garcia Satorras, Clement Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International Conference on Machine Learning*, pp. 8867–8887. PMLR, 2022.

Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi Jaakkola. Torsional diffusion for molecular conformer generation. *arXiv preprint arXiv:2206.01729*, 2022.

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.

R. Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997.

Jayoung Kim, Jinsung Jeon, Jaehoon Lee, Jihyeon Hyeong, and Noseong Park. Oct-gan: Neural ode-based conditional tabular gans. In *Proceedings of the Web Conference 2021*, pp. 1506–1515, 2021.

Jayoung Kim, Chaejeong Lee, Yehjin Shin, Sewon Park, Minjung Kim, Noseong Park, and Jihoon Cho. Sos: Score-based oversampling for tabular data. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 762–772, 2022.

Ron Kohavi. Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In *KDD*, 1996.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

Haoying Li, Yifan Yang, Meng Chang, Huajun Feng, Zhihai Xu, Qi Li, and Yueting Chen. Srdiff: Single image super-resolution with diffusion probabilistic models. 2021.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *arXiv preprint arXiv:2205.14217*, 2022.

Renata C. B. Madeo, Clodoaldo Ap. M. Lima, and Sarajane Marques Peres. Gesture unit segmentation using support vector machines: segmenting gestures from rest positions. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC*, 2013.

Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Image synthesis and editing with stochastic differential equations. 2021.

Prafulla Nichol, Alex & Dhariwal. Improved denoising diffusion probabilistic models. *ICML*, 2021.

Richard Nock and Mathieu Guillame-Bert. Generative trees: Adversarial and copycat. *ICML*, 2022.

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

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. 2021.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

Kamaljot Singh, Ranjeet Kaur Sandhu, and Dinesh Kumar. Comment volume prediction using neural networks and decision trees. In *IEEE UKSim-AMSS 17th International Conference on Computer Modelling and Simulation, UKSim*, 2015.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *NeurIPS*, 2020.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. 2021.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.

Amirsina Torfi, Edward A Fox, and Chandan K Reddy. Differentially private synthetic medical data generation using convolutional gans. *Information Sciences*, 586:485–500, 2022.

Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luís Torgo. Openml: networked science in machine learning. *arXiv*, 1407.7722v1, 2014.

Bingyang Wen, Yupeng Cao, Fan Yang, Koduvayur Subbalakshmi, and Rajarathnam Chandramouli. Causal-tgan: Modeling tabular data using causally-aware gan. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2022.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in Neural Information Processing Systems*, 32, 2019.

Yishuo Zhang, Nayyar A Zaidi, Jiahui Zhou, and Gang Li. Ganblr: a tabular data generation model. In *2021 IEEE International Conference on Data Mining (ICDM)*, pp. 181–190. IEEE, 2021.

Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. Ctab-gan: Effective table data synthesizing. In *Asian Conference on Machine Learning*, pp. 97–112. PMLR, 2021.

Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. Ctab-gan+: Enhancing tabular data synthesis. *arXiv preprint arXiv:2204.00401*, 2022.

# APPENDIX

## A MLP EVALUATION AND TUNING

Here, we show that tuning the hyperparameters using the CatBoost guidance results in the TabDDPM models that produce synthetics that is also optimal for other classifiers/regressors. The results for a subset of datasets are presented on Table 6. The methods denoted with ”-CB” and ”-MLP” denote the CatBoost guidance and different types of evaluation (CatBoost and MLP, respectively). The ”-MLP-tune” suffix stands for the MLP guidance tuning and MLP evaluation.

| | AB ($R2$) | AD ($F1$) | BU ($F1$) | CA ($R2$) | CAR ($F1$) | CH ($F1$) | DE ($F1$) | DI ($F1$) |
|---|---|---|---|---|---|---|---|---|
| TabDDPM-CB | $0.550_{\pm.010}$ | $0.795_{\pm.001}$ | $0.906_{\pm.003}$ | $0.836_{\pm.002}$ | $0.737_{\pm.001}$ | $0.755_{\pm.006}$ | $0.691_{\pm.004}$ | $0.740_{\pm.020}$ |
| Real-CB | $0.556_{\pm.004}$ | $0.815_{\pm.002}$ | $0.906_{\pm.002}$ | $0.857_{\pm.001}$ | $0.738_{\pm.001}$ | $0.740_{\pm.009}$ | $0.688_{\pm.003}$ | $0.785_{\pm.013}$ |
| TabDDPM-MLP | $0.569_{\pm.010}$ | $0.794_{\pm.002}$ | $0.903_{\pm.003}$ | $0.809_{\pm.003}$ | $0.737_{\pm.001}$ | $0.750_{\pm.005}$ | $0.679_{\pm.008}$ | $0.754_{\pm.020}$ |
| Real-MLP | $0.581_{\pm.005}$ | $0.795_{\pm.001}$ | $0.905_{\pm.003}$ | $0.808_{\pm.002}$ | $0.739_{\pm.001}$ | $0.741_{\pm.006}$ | $0.688_{\pm.004}$ | $0.754_{\pm.017}$ |
| TabDDPM-MLP-tune | $0.559_{\pm.009}$ | $0.792_{\pm.002}$ | $0.901_{\pm.003}$ | $0.803_{\pm.004}$ | $0.737_{\pm.001}$ | $0.749_{\pm.006}$ | $0.674_{\pm.013}$ | $0.741_{\pm.018}$ |

| | FB ($R2$) | GE ($F1$) | HI ($F1$) | HO ($R2$) | IN ($R2$) | KI ($R2$) | MI ($F1$) | WI ($F1$) |
|---|---|---|---|---|---|---|---|---|
| TabDDPM-CB | $0.713_{\pm.002}$ | $0.597_{\pm.006}$ | $0.722_{\pm.001}$ | $0.677_{\pm.010}$ | $0.809_{\pm.002}$ | $0.833_{\pm.014}$ | $0.936_{\pm.001}$ | $0.904_{\pm.009}$ |
| Real-CB | $0.837_{\pm.001}$ | $0.636_{\pm.007}$ | $0.724_{\pm.001}$ | $0.662_{\pm.003}$ | $0.814_{\pm.001}$ | $0.907_{\pm.002}$ | $0.934_{\pm.000}$ | $0.898_{\pm.006}$ |
| TabDDPM-MLP | – | $0.595_{\pm.006}$ | $0.717_{\pm.002}$ | $0.643_{\pm.010}$ | $0.794_{\pm.008}$ | $0.804_{\pm.015}$ | $0.938_{\pm.001}$ | $0.921_{\pm.006}$ |
| Real-MLP | – | $0.607_{\pm.007}$ | $0.717_{\pm.002}$ | $0.614_{\pm.006}$ | $0.800_{\pm.003}$ | $0.882_{\pm.004}$ | $0.936_{\pm.001}$ | $0.905_{\pm.006}$ |
| TabDDPM-MLP-tune | – | – | – | $0.626_{\pm.009}$ | $0.800_{\pm.004}$ | $0.799_{\pm.018}$ | – | $0.914_{\pm.006}$ |

Table 6: ML utility score with MLP evaluation and MLP tuning compared with CatBoost evaluation and CatBoost tuning.

## B HYPERPARAMETERS SEARCH SPACES

| Parameter | Distribution |
|---|---|
| Max depth | $\mathrm{UniformInt}[3, 10]$ |
| Learning rate | $\mathrm{LogUniform}[1e\text{-}5, 1]$ |
| Bagging temperature | $\mathrm{Uniform}[0, 1]$ |
| L2 leaf reg | $\mathrm{LogUniform}[1, 10]$ |
| Leaf estimation iterations | $\mathrm{UniformInt}[1, 10]$ |
| Number of tuning trials | 100 |

Table 7: CatBoost hyperparameters space from Gorishniy et al. (2021)

| Parameter | Distribution |
|---|---|
| # Layers | $\mathrm{UniformInt}[1, 8]$ |
| Layer size | $\mathrm{Int}\{64, 128, 256, 512, 1024\}$ |
| Dropout | $\{0, \mathrm{Uniform}[0, 0.5]\}$ |
| Learning rate | $\mathrm{LogUniform}[1e\text{-}5, 1e\text{-}2]$ |
| Weight decay | $\{0, \mathrm{LogUniform}[1e\text{-}6, 1e\text{-}3]\}$ |
| Number of tuning trials | 100 |

Table 8: MLP hyperparameters space from Gorishniy et al. (2021)

| Parameter | Distribution |
|---|---|
| k_neighbours | $\mathrm{Int}[5, 20]$ |
| $\lambda_{range}$ | $\mathrm{Float}[0, 1]$ |
| Proportion of samples | $\mathrm{Float}\{0.25, 0.5, 1, 2, 4, 8\}$ |
| Number of tuning trials | 50 |

Table 9: SMOTE hyperparameters search space. $\lambda_{range}$ denotes the range of interpolation coefficient to sample from

| Parameter | Distribution |
|---|---|
| # claassif. layers | $\mathrm{UniformInt}[1, 4]$ |
| Classif. layer size | $\mathrm{Int}\{64, 128, 256\}$ |
| Training iterations | $\mathrm{Cat}\{1000, 5000, 10000\}$ |
| Batch Size | $\mathrm{Int}\{512, 1024, 2048\}$ |
| random_dim | $\mathrm{Int}\{16, 32, 64, 128\}$ |
| num_channels | $\mathrm{Int}\{16, 32, 64\}$ |
| Proportion of samples | $\mathrm{Float}\{0.25, 0.5, 1, 2, 4, 8\}$ |
| Number of tuning trials | 35 |

Table 10: CTABGAN and CTABGAN+ hyperparameters search space. See an official implementation[2]

| Parameter | Distribution |
|---|---|
| # claassif. layers | $\mathrm{UniformInt}[1, 6]$ |
| Classif. layer size | $\mathrm{Int}\{64, 128, 256, 512\}$ |
| Training iterations | $\mathrm{Cat}\{5000, 20000, 30000\}$ |
| Batch Size | $\mathrm{Cat}\{456, 4096\}$ |
| embedding_dim | $\mathrm{Int}\{16, 32, 64, 128, 256, 512, 1024\}$ |
| loss factor | $\mathrm{LogUniform}[0.01, 10]$ |
| Proportion of samples | $\mathrm{Float}\{0.25, 0.5, 1, 2, 4, 8\}$ |
| Number of tuning trials | 50 |

Table 11: TVAE hyperparameters search space. See an official implementation[3]

## C Datasets

We used the following datasets:

- Abalone (OpenML)
- Adult (income estimation, Kohavi (1996))
- Buddy (Kaggle)
- California Housing (real estate data, Kelley Pace & Barry (1997))
- Cardiovascular Disease dataset (Kaggle)
- Churn Modeling (Kaggle)
- Diabetes (OpenML)
- Facebook Comments (Singh et al. (2015))
- Gesture Phase Prediction (Madeo et al. (2013))

[3] https://github.com/Team-TUD/CTAB-GAN-Plus

[3] https://github.com/sdv-dev/CTGAN

- Higgs (simulated physical particles, Baldi et al. (2014); we use the version with 98K samples available at the OpenML repository Vanschoren et al. (2014))
- House 16H (OpenML)
- Insurance (Kaggle)
- King (Kaggle)
- MiniBooNE (OpenML)
- Wilt (OpenML)