Mitigating Class Imbalance in Tabular Data through Neural Network-based Synthetic Data Generation: A Comprehensive Survey and Library

Preprint, compiled August 9, 2025

Omar A. Mures [1,4*], **Javier Taibo** [1], **Emilio J. Padrón** [2,4], **and Jose A. Iglesias-Guitian** [3,4]

[1]Computer Graphics & Visual Computing (XLab), Department of Civil Engineering, Universidade da Coruña
[2]Computer Graphics & Visual Computing (XLab), Department of Computer Engineering, Universidade da Coruña
[3]Computer Graphics & Visual Computing (XLab), Department of Computer Science and IT, Universidade da Coruña
[4]Centre for ICT Research (CITIC), Universidade da Coruña

Abstract

Imbalanced datasets often bias downstream models towards favoring majority classes, posing a critical challenge in deep learning, where extensive data is pivotal for optimal performance. Traditional solutions, such as classical data augmentation, often struggle with nuanced data traits and lack adaptability. The emergence of deep learning techniques like Auto Encoders (AEs), Generative Adversarial Networks (GANs), Diffusion Models (DMs), and Large Language Models (LLMs) opens promising avenues for addressing class imbalance through synthetic data generation. This paper presents a comprehensive survey of generative AI techniques for mitigating class imbalance in tabular datasets. These methods have the potential to improve the performance and efficiency of data-driven models across multiple domains. We evaluate their effectiveness in applications like handball play classification, income level prediction, and used car evaluation. We not only assess their efficacy in these real-world applications but also introduce computational efficiency tests, an often-overlooked aspect in this field. In addition to the survey, we present 'GenTab,' a synthetic tabular data generation library to facilitate the implementation and evaluation of the discussed approaches. GenTab is accessible on GitHub and offers a user-friendly framework for practitioners to leverage cutting-edge generative models for synthetic tabular dataset creation or augmentation.

1 Introduction

The widespread adoption of Machine Learning (ML) and Deep Learning (DL) techniques across diverse research fields has underscored the persistent challenges posed by 'class imbalance' [1, 2, 3, 4]. Class imbalance refers to a scenario in a dataset where one or more classes significantly outnumber others. Within this context, classes with higher representation are termed 'majority classes', whereas those with less representation are deemed 'minority classes'. This imbalance can impact the performance and accuracy of downstream models, making it a crucial consideration in dataset preparation and model training. This issue becomes particularly critical when minority classes, despite being limited in number, hold significant importance. This problem is prevalent across various domains, such as financial data analysis [5], where it can have severe consequences. Imbalanced datasets can contribute to discriminatory lending practices or biases when assessing creditworthiness. In the domain of indoor invasion sports, such as handball, penalties are rare but critical game situations that pose substantial challenges to automated production systems [6]. Examples like these highlight the need for robust solutions to address class imbalance in diverse and practical scenarios.

Deep learning techniques, often characterized by their 'black-box' modeling of relationships between input and output variables, encounter unique challenges when confronted with imbalanced data. This complexity is particularly relevant given how neural networks update their weights, often favoring the majority classes, as discussed in [7]. Despite the numerous proposed approaches to mitigate this issue, addressing class imbalance remains an open problem, with many existing solutions yielding unsatisfactory results, as evidenced by [8]. Furthermore, neural networks may learn random replications of the most frequent samples, further complicating their training process. Advancements in deep learning, such as Auto Encoders (AEs) [9], Generative Adversarial Networks (GANs) [10], Diffusion Models (DMs) [11], and Large Language Models (LLMs) [12], have garnered significant interest and present clear opportunities for synthetic data generation.

Deep learning architectures surpass the limitations of traditional data augmentation techniques (e.g., data sampling or replication) by accounting for data space density and capturing complex relationships within the original dataset. As such, they are considered 'global scope' models. In contrast, random oversampling methods such as SMOTE [13] and ADASYN [14] create new samples using nearest-neighbor interpolation, assuming that the space between existing samples is also representative of the minority class, which might be a strong assumption, i.e., a new class sample could be located outside of the observed minority class manifold. While effective in addressing class imbalance in lower-dimensional problems, their efficacy diminishes in high-dimensional space, as demonstrated in [15]. These 'local scope' models, focused on single observations or close neighbors, may generate unrealistic and noisy synthetic samples that lie outside the actual minority class manifold and struggle to capture nonlinear relationships. Other classical approaches, like Gaussian Copula [16], rely on predefined statistical assumptions (e.g., marginal distributions), which often fail to model high-

dimensional, nonlinear real-world tabular data distributions.

As outlined above, numerous methods exist for synthetic tabular data generation. This work aims to provide an in-depth survey of key approaches, with a primary focus on AEs, GANs, DMs, and LLMs. These generative methods are increasingly essential for tasks such as data augmentation, class imbalance mitigation, dataset anonymization, and resampling. To complement this survey, we introduce an open-source library that lowers entry barriers by providing access to state-of-the-art techniques, enabling researchers to easily apply them to their own datasets. The library is also designed for extensibility, allowing seamless integration of new data generation models.

What sets our survey apart from others While existing surveys have explored various architectures for synthetic tabular data generation, such as probabilistic models [17], AEs [18], or GANs [19], our work addresses the gap concerning newer neural network-based approaches. We provide a comprehensive and up-to-date coverage of the latest advancements in this area, focusing particularly on advanced architectures such as DMs and LLMs, which remain underexplored in previous surveys [20, 21, 22, 23]. This effort builds on and extends the taxonomy provided by prior work [20], incorporating additional state-of-the-art approaches and offering a more detailed classification of these emerging methods. Unlike surveys that primarily focus on theoretical discussions, we complement our analysis with extensive testing of the surveyed techniques across diverse datasets. Furthermore, we introduce a dedicated synthetic tabular data generation library which was used in our experiments. This library was designed to make state-of-the-art methods more accessible and reproducible. By combining theoretical insights with practical tools, our survey bridges a significant gap in the current literature, providing researchers with the knowledge and resources needed to leverage these powerful techniques effectively. In summary, our contributions are threefold:

- **Survey and taxonomy:** We provide an in-depth survey of the most relevant neural network-based architectures for synthetic tabular data generation and extend the taxonomy in [20] to include state-of-the-art approaches like DMs and LLMs.
- **Synthetic tabular data generation library:** We have developed an open-source library, 'GenTab', designed to easily generate, tune, and evaluate synthetic data using advanced methods, making these techniques accessible to a wide range of users and encouraging reproducibility.
- **Benchmarking:** We conduct benchmarking of selected state-of-the-art methods across diverse datasets to assess performance and applicability, including a computational efficiency analysis, an important aspect often overlooked in the existing literature.

2 BACKGROUND

Our study delves into the fundamental concepts, objectives, and motivations behind synthetic tabular data generation. Synthetic data refers to artificially created samples that mimic the characteristics of an original dataset using generative models trained on existing data.

Buying	Maint	Doors	Persons	Lug Boot	Safety	Class
vhigh	vhigh	2	2	small	low	unacc
vhigh	med	2	4	big	med	accept
med	low	2	4	small	high	good
med	low	2	more	big	high	vgood
med	low	2	more	small	low	unacc

Figure 1: Example from one of the chosen datasets, UCI Car Evaluation [24]. Columns represent car properties and rows car instances. *Buying* represents its overall price, *Maint* its maintenance price, *Doors* its door number, *Persons* its capacity, *Lug Boot* the luggage boot size, *Safety* the estimated safety, and *Class* the evaluation level of the car.

2.1 Synthetic Tabular Data Generation

Although the 'tabular data' concept is quite common, providing a brief definition is essential. Tabular data usually comprises rows and columns. Columns define attributes relevant to a particular domain, while rows represent individual samples. Throughout the text, we may interchangeably refer to them as rows, samples, or observations. For example, in used car evaluation (see Fig. 1), each row represents a specific car, and the columns detail car attributes, such as its price, number of doors, capacity, and condition. It is important to note that multiple rows may contain identical information in a tabular dataset. Sometimes, an attribute may be missing from an instance, and a placeholder or invariant value is often used to indicate the lack of data.

We assume access to N labeled rows from a source dataset, denoted as $D = \{(X_i, Y_i)\}_{i=1}^{N}$. In this representation, X_i signifies an observation (comprising either numeric or categorical features), while Y_i represents its corresponding class or label. In many instances, tabular data exhibits a relationship between Y_i (the dependent variable), and the rest of the columns, X_i (the independent variables). These independent variables explain changes or variations in the dependent variable. While such a relationship is optional, our survey focuses primarily on works where the ultimate goal involves understanding or leveraging this relationship. This focus aligns with using tabular data in supervised learning scenarios, where the interaction between these variable types is critical.

In these cases, the structure and nature of the data are pivotal in developing effective models that can accurately capture and utilize these relationships for making informed predictions or decisions. The ultimate goal in our case study is training a classification model, $f : \mathcal{X} \rightarrow \mathcal{Y}$, to accurately classify unseen data. Our focus is on generating an enhanced synthetic training set $D^S = \{(X_i^S, Y_i^S)\}_{i=1}^{N'}$ utilizing synthetic data generation techniques. This involves employing a generator function $f_{gen}(\cdot; \rho) = (X^S, Y^S)$, where ρ represents the generator hyperparameters, and $(X^S, Y^S) \in D^S$ a synthetically generated row (being X^S its features and Y^S its corresponding class). The generator produces a synthetic dataset D^S of arbitrary length, crafted to not only mimic the original data but also to enhance the training process of the classification model. Key issues such as class imbalance and feature representation are specifically addressed in D^S.

Table 1: Summary of neural network-based synthetic data generators, highlighting their key characteristics and differences.

	Classical		Neural Network-based			
	Randomized	**Probabilistic**	**AE**	**GAN**	**DM**	**LLM**
Nonlinear Pattern Support	○	◐	●	●	●	●
Dimensionality Support	○	◐	●	●	●	●
Noise Robustness	○	◐	●	●	●	●
Diversity	○	○	◐	●	●	●
Computational Efficiency	●	◐	◐	◐	○	○
Fidelity	◐	◐	◐	◐	●	◐
Privacy	○	●	◐	●	◐	○
Oversampling Performance	◐	◐	●	◐	●	○

○ = Low ◐ = Moderate ● = High

Evaluations have a white background when based on theoretical assumptions (Sec. III), gray backgrounds signify they are based on experimental data (Sec. V).
Elements in blue can be clicked to navigate to the relevant part in the paper.

2.2 *Case Studies*

Our survey analyzes two of the most significant applications of synthetic tabular data generation: oversampling for mitigating class imbalance and dataset anonymization for privacy.

Class Imbalance This survey mainly focuses on the challenge of class imbalance, specifically addressing scenarios where the dependent variable presents a significant imbalance in its distribution. In these situations, the distribution of categories or classes within the dependent variable is uneven, often leading to a skew in the dataset where one or more classes are underrepresented compared to others. This imbalance has the potential to negatively affect downstream models, particularly in supervised learning tasks, where the model may suffer from bias towards the majority class, failing to recognize or predict instances of the minority class adequately. One of the most common approaches for addressing this issue is oversampling [25]. This technique generates new observations for minority classes achieving a more balanced distribution. Our survey aims to explore and highlight neural network-based methodologies, techniques, and approaches that mitigate class imbalance issues in tabular data.

Privacy Although not the main focus of our survey, synthetic tabular data generation also offers a compelling solution for dataset anonymization, addressing the ongoing challenge of balancing data utility with privacy concerns [26]. This approach seeks to create synthetic datasets that minimize re-identification risks by ensuring that individual samples cannot be linked to actual dataset records. The central goal is to produce an artificial dataset that eliminates identifiable samples, safeguarding privacy while keeping the downstream utility of the data intact. This anonymization process requires generative techniques capable of accurately modeling the original dataset and ensuring sufficient distinctiveness in synthetic samples to prevent re-identification. This is particularly crucial in healthcare, finance, or social sciences, where sensitive data requires analytic integrity without compromising confidential information [27].

3 NETWORK-BASED SYNTHETIC TABULAR DATA GENERATION

In [20], a general taxonomy for synthetic tabular data generation is presented. Our survey extends this taxonomy by providing additional details about the architectures and mechanisms for data generation, incorporating previously underexplored approaches. For a comprehensive summary of the surveyed architectures and their differences compared to classical approaches, see Table 1. To facilitate navigation and provide details about the reasoning behind the chosen evaluations, highlighted elements in the table link to the corresponding sections of the paper where longer form explanations are provided. We focus specifically on generative models, categorized under network-based approaches. These approaches typically generate samples in latent or input space that yield observations virtually indistinguishable from those in the original dataset.

Conversely, traditional approaches rely on information from individual samples, nearby neighbors, or predefined parametric distributions. These methods can be sensitive to outliers and noise in minority samples, which may lead to the generation of synthetic samples that also exhibit these issues and lie outside the true minority class manifold. The lack of inherent regularization mechanisms in many traditional methods exacerbates this issue, increasing the risk of overfitting. Moreover, in high-dimensional spaces, nearest-neighbor searches become less reliable, and linear interpolations less meaningful, limiting the variability and reliability of generated synthetic samples. Traditional methods also struggle to accurately represent nonlinear relationships due to their reliance on underlying assumptions of linearity or simple parametric distributions, which can fail to capture the non-monotonic dependencies that can exist between features in real-world datasets, thus limiting the diversity and quality of generated samples.

These limitations contrast with deep generative models (such as GANs, VAEs, DMs, and LLMs) which have the capacity, via neural networks, to learn much more complex, nonlinear relationships and approximate the true underlying data distribution more faithfully. As a result, they can generate more diverse

Table 2: Classification of the selected synthetic data generation methods, categorizing them within their corresponding architecture types. Each method is detailed in terms of its specific architecture type (used technique or approach), architecture (broader data augmentation type), level (stage of the ML pipeline in which the generative process is applied), data space (type of data representation the model applies to), and scope (extent of utilization of the underlying dataset distribution properties). This classification aligns and extends the categorization practices in [20].

Algorithm	Type	Architecture	Level	Data space	Scope
SMOTE [13]	Linear	Randomized	External	Input	Local
ADASYN [14]	Linear	Randomized	External	Input	Local
GaussianCopula [16]	PDF	Probabilistic	External	Latent	Global
TVAE [28]	AE	Network	External	Latent	Global
CTGAN [28]	GAN	Network	External	Latent	Global
CTAB-GAN [29]	GAN	Network	External	Latent	Global
CTAB-GAN+ [30]	GAN	Network	External	Latent	Global
CopulaGAN [16, 28]	PDF+GAN	Probabilistic+Network	External	Latent	Global
ForestDiffusion [31]	DM	Network	External	Latent	Global
AutoDiffusion [32]	AE+DM	Network	External	Latent	Global
GReaT [33]	LLM	Network	External	Input+Latent	Global
Tabula [34]	LLM	Network	External	Input+Latent	Global

and realistic synthetic samples consistent with the overall data distribution, potentially offering better coverage of the minority class manifold. Given these considerations, our expanded taxonomy encompasses the most relevant neural network-based architectures. Table 2 illustrates the chosen tabular data generation methods and their classification within the expanded taxonomy.

3.1 Auto Encoders

AEs [35] are specialized neural networks that focus on learning a latent representation of the input data, which is reproduced in the output layer. These networks are trained in an unsupervised manner and typically consist of three elements, the encoder, the 'bottleneck', and the decoder (see Fig. 2). The encoder maps input vectors into hidden representations in latent space, compressing the input into a lower-dimensional form. This latent space is a compact, encoded representation, that captures its key features and patterns. Following the bottleneck is the decoder, which regenerates the input data from its latent space representation, attempting to map the compressed data back to the original output space. Once the latent representation is obtained, it can be perturbed or manipulated to generate new data samples. This capability makes AEs particularly useful for tasks like dimensionality reduction, feature extraction, denoising, and, in our case, synthetic tabular data generation.

A key advancement in AEs was the development of the Variational Auto Encoder (VAE), introduced in [36]. VAEs employ variational inference for continuous latent space representation learning, facilitating smooth interpolation between training data points. The variational aspect in VAEs forces the model to learn a distribution in latent space, not just point estimates, making it less susceptible to overfitting individual minority samples. Sampling from a learned distribution in latent space, VAEs can generate a wider variety of synthetic samples, covering more of the minority class manifold and providing increased diversity, which is crucial for improving downstream classifier performance.

The adaptation of VAEs for tabular data generation, and particularly the TVAE [28] model, stands out as one of the key developments for employing purely AE-based architectures in synthetic tabular data generation tasks. TVAE adapts the VAE framework to handle tabular data through specific pre-processing techniques. This includes representing categorical values as one-hot vectors and employing 'mode-specific' normalization. The normalization process uses a Variational Gaussian Mixture (VGM) model, as detailed in [37], which is particularly effective when dealing with numerical columns that have complex distributions. In this representation, each value is encoded in a one-hot vector denoting its mode, along with a number representing the value within that mode. Furthermore, TVAE, structurally similar to a VAE, comprises two neural networks that are jointly trained using evidence lower-bound (ELBO) loss [36]. The KL divergence term in their loss function acts as a regularizer, encouraging the latent space to be smooth and well-behaved, mitigating overfitting to the training data (including noisy minority samples), a common problem in local oversampling methods.

AEs, and particularly VAEs, offer a promising approach for tackling class imbalance. They have shown strong results in similar areas, such as synthetic data generation for imbalanced learning [38]. In these applications, they often surpass traditional oversampling techniques that simply create copies or slight variations of existing minority class samples or predefined distributions. The encoder and decoder networks can learn highly nonlinear mappings, allowing AEs to capture complex dependencies between features and generate synthetic data that reflects these nonlinearities. This capacity to capture complex, nonlinear relationships between features enhances their ability to create diverse yet representative synthetic data.

For those seeking more comprehensive information on AEs, their development, applications, and underlying principles, refer

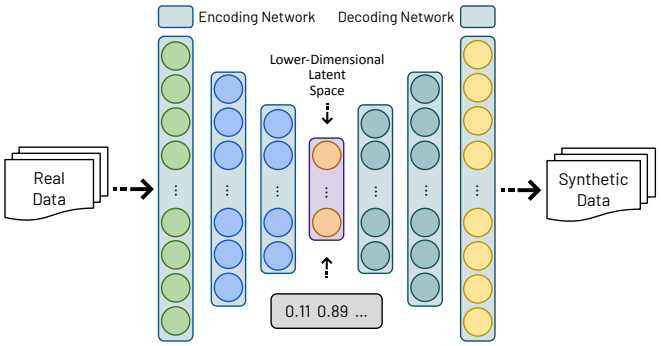

Figure 2: Illustration detailing the inner-workings of Auto Encoders (AEs). It shows two key components in this architecture, the encoder and the decoder. The encoder (blue) reduces the dimensionality of the input, compressing it down to a latent space representation. Following this process, the decoder (teal) recreates the input from its latent space representation back to output space, aiming to replicate the original input data as closely as possible. We can introduce synthetic latent vectors into the trained decoding network to create new observations that mimic real data.

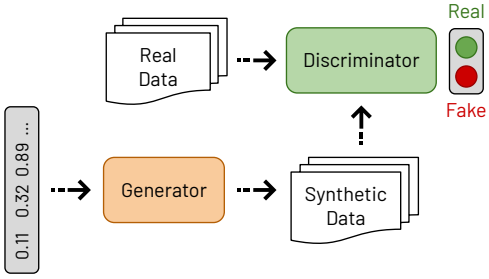

Figure 3: Diagram showcasing the functioning of Generative Adversarial Networks (GANs), highlighting the competition between the generator (orange) and the discriminator (green). The generator creates new observations from noise, while the discriminator, receiving both real and fake data, learns to distinguish between them. Over time, the generator aims to produce higher-quality samples, making it increasingly difficult for the discriminator to tell whether the data is real or synthetic.

to [39]. For a survey more in line with our case study, readers can peruse [18], exploring their use for missing data imputation.

3.2 *Generative Adversarial Networks*

Since their inception, GANs [40] have enticed significant research interest. GANs engage two neural networks in a contest: one network functions as a generator and the other as a discriminator (see Fig. 3). The fundamental concept of GANs lies in their indirect training approach, primarily through the discriminator, a network tasked with evaluating the realism of inputs. The generative network, trained to reconstruct data from a latent space vector (noise), creates candidate data, while the discriminative network assesses these candidates against the actual data distribution. The generative network aims to trick the discriminator into classifying its synthetic output as part of the real data distribution. This adversarial learning process makes them highly effective when generating synthetic observations that are representative of the underlying data distribution. By incentivizing the generator to reproduce the intricate patterns present in the real data distribution, such as nonlinearities, GANs can create synthetic data that preserves not only individual column properties but also complex feature interactions and conditional dependencies. Moreover, the generator learns to map a low-dimensional noise vector to points within the high-dimensional feature space data manifold, bypassing the need to model the probability density across the entire high-dimensional space, therefore mitigating the curse of dimensionality. While GANs are proficient in the synthetic data generation task, they encounter several challenges, including vanishing gradients, mode collapse, and difficulties converging, as outlined in [41].

Historically, GANs have been predominantly utilized in computer vision class imbalance problems [42], but their potential applications extend beyond image data. Early GAN implementations, lacking mechanisms for specifying the desired output classes, were limited in their ability to address class imbalance problems through the targeted generation of minority class samples. Significant efforts have been directed towards resolving this issue, notably CGAN [43] and BAGAN [44]. While being slight modifications of the original GAN architecture, these architectures incorporate a key feature: they condition the network with additional information during training, such as class labels, allowing for the targeted generation of minority class samples. Another noteworthy development for our case study is the conditional Wasserstein GAN (WGAN) oversampling method in [45]. This approach utilizes the CGAN structure, incorporating the WGAN gradient penalty (WGAN-GP) loss function and an Auxiliary Classifier (AC) [46] loss to improve minority sample generation and create not only plausible but truly recognizable synthetic data points. Furthermore, GWGAN-GP [47] employs Gaussian distribution labels to disperse synthetic examples, reducing redundancy and increasing diversity compared to local methods.

A significant contribution to GAN-based methods is CTGAN [28], a novel adaptation of GANs explicitly tailored for generating realistic synthetic tabular data. CTGAN introduces a new conditional generator, which plays a critical role in managing discrete and imbalanced columns in the data. By conditioning the generation process on specific categorical variables, it ensures the realistic representation of both common and rare categories while preserving complex data relationships missed by local methods. The training-by-sampling approach in CTGAN is designed to effectively sample and model different distributions present in tabular data. During training, categories are sampled according to their log-frequency, ensuring rare categories are equally explored. This prevents the generator from favoring majority classes and mitigates mode collapse. CopulaGAN [16, 28], is a modification of CTGAN that integrates classical statistical methods with GAN-based approaches. It is a hybrid that combines the Gaussian Copula [48], a well-known statistical method for learning the overall distribution of real data, with the architecture of CTGAN.

CTAB-GAN [29] is another iteration of the CGAN methodology, with enhanced capabilities to model varied data types, which include combinations of continuous and categorical variables. Several elements are introduced to refine the CGAN framework. These include information loss and classification

loss for addressing class imbalance and long tail issues. The information loss helps ensure that the synthetic data retains key characteristics of the original dataset (i.e., mean and standard deviation), while the classification loss aids in maintaining semantic integrity penalizing samples with incorrect combinations of values. Another novel aspect of CTAB-GAN is its unique conditional vector design. This conditional vector is specifically engineered to encode diverse data types and variables with a skewed distribution (a common occurrence in imbalanced datasets).

CTAB-GAN+ [30], an enhanced version of the CTAB-GAN architecture, offers several improvements when generating synthetic data for imbalanced datasets. This improved model incorporates downstream losses into the conditional GAN framework, specifically aimed at increasing the performance of synthetic data in downstream tasks. To address class imbalance problems, it utilizes two techniques from CTGAN: a conditional generator and training-by-sampling. In addition, it uses Wasserstein loss [49] with gradient penalty, a modification designed to improve training convergence and stability, making this model less susceptible to noise in minority samples. CTAB-GAN+ also introduces novel encoders that are specifically tailored to handle mixed continuous or categorical variable types, as well as variables with skewed distributions.

For a more detailed exploration of how GANs can be used to mitigate class imbalance problems in tabular data, a comprehensive review is available in [50].

3.3 *Diffusion Models*

The field of generative AI has recently witnessed a rise in interest regarding diffusion models, owing to their performance, which often matches or surpasses that of state-of-the-art GANs. These models, a category of probabilistic generative models, introduce random noise to observations, subsequently learning to revert this process and regenerate them from that noise (see Fig. 4). This process allows DMs to generate new synthetic samples and enables them to progressively build internal structures that accurately model input data distributions. To effectively predict the noise at each step, the network must implicitly learn the underlying structure of the data manifold, including its nonlinearities, even when partially hidden by noise. This step-by-step denoising process breaks down complexity, allowing DMs to gradually reconstruct high-dimensional samples that adhere to the learned data structure. As a result, DMs have been highly successful in generating high-quality and realistic images. They are also known for their ability to create diverse outputs with intricate details, which makes them suitable for our case study. Although, similarly to GANS, DMs have primarily been applied to address class imbalance issues in image datasets [51, 52], their success in these applications in terms of fidelity and diversity, suggests their potential for mitigating class imbalance issues in tabular datasets. Diffusion model research is generally based on three primary formulations: Denoising Diffusion Probabilistic Models (DDPMs) [53, 11], Score-based Models (SGMs) [54], and Stochastic Differential Equations (SDEs) [55].

DDPMs, rooted in the principles of non-equilibrium thermodynamics, employ a dual Markov chain of diffusion stages (forward and backward). The forward process comprises the diffusion of data with pre-determined noise (i.e., Gaussian noise), while the

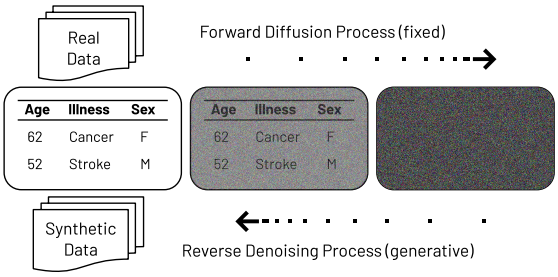

Figure 4: Illustration of the operation of Diffusion Models (DMs). The figure depicts their two main processes: the forward diffusion process, where the real input is gradually corrupted by adding noise, and the backward diffusion process, where the model learns to reverse the noise addition to generate new samples.

reverse process employs neural networks to eliminate noise and recover the original data sequentially. While DDPMs have been typically applied to continuous data types like audio and images through Gaussian diffusion, their application can be extended to other domains through multinomial diffusion, as discussed in [56]. Multinomial diffusion is specifically designed to handle categorical data, providing a method for applying diffusion model principles to discrete data types, allowing these models to be applied to the task at hand, synthetic tabular data generation.

TabDDPM [57] is a prime example of how DDPMs can be utilized to generate synthetic tabular data in imbalanced scenarios. This model employs a distinct diffusion process for each feature type: Gaussian diffusion for numerical features and multinomial diffusion for categorical features. Multinomial diffusion operates by gradually adding uniform categorical noise during the forward process and reversing it through iterative denoising. Each categorical feature undergoes an independent forward diffusion process, isolating its noise dynamics from more frequent features, thereby helping to avoid mode collapse. Additionally, it uses a class-conditional design, which enables explicit modeling of minority classes by conditioning on class labels.

SGMs utilize a score function to learn the logarithm of the gradient for the Probability Density Function (PDF) inherent in the actual input data. High-dimensional data and deep neural network contexts, make directly obtaining this function not feasible. Various approaches have been developed to overcome this challenge. Score-Matching, as introduced in [58], focuses on estimating the probability density function score, which tries to address many of the problems in likelihood-based methods. Denoising Score Matching, proposed in [54], extends this concept by training the model to predict the score of the noiseless data distribution, obtaining a strong prior for the clean signal. Sliced Score Matching, detailed in [59], further simplifies the score-matching task by using a one-dimensional data distribution, reducing complexity and enhancing computational efficiency for high-dimensional data. For a further look into SGMs, the reader can peruse [60, 61].

SDE based models share similar objectives with both SGMs and DDPMs.[55] extends the number of noise scales, finite in traditional SGMs and DDPMs, to infinity by applying SDEs. This expansion of noise levels allows for more continuous and precise modeling, enhancing the effectiveness of generative mod-

els. During training, instead of directly approximating the score functions (computationally infeasible), transition probabilities are estimated. After the training process, samples can be generated using several methods such as Euler-Maruyama (EM), Prediction-Correction (PC), or probability flow Ordinary Differential Equations (ODEs).

Diving specifically into our problem at hand, dataset imbalance, Score-based Over Sampling (SOS) [62] is the first work introducing a score-based tabular data oversampling method. This technique is reminiscent of a style transfer method, as it transforms samples from majority classes by adding controlled noise through a forward SDE into synthetic minority class samples (using a reverse SDE to denoise them). This process is guided by a class-conditioned score function that ensures the generated samples resemble the minority class distribution. This approach avoids generating synthetic samples in isolation and instead leverages existing majority class data to inform minority class synthesis, improving boundary alignment between classes. Finally, a class-conditional fine-tuning scheme can be optionally applied to improve minority class sampling performance.

STaSy [63], employing SGMs for tabular data synthesis, addresses the challenges of directly applying SDEs to this domain. The authors highlight the difficulty in learning joint probabilities of columns when using standard SDEs, to overcome this, they introduce dataset-dependent Multilayer Perceptron (MLP) residual blocks. Furthermore, they propose a novel training strategy combining self-paced learning (SPL) with denoising score matching. SPL dynamically adjusts the training process by prioritizing easier samples with low training losses initially and gradually incorporating the rest of the data. This approach helps mitigate uneven loss distributions often observed in imbalanced datasets. For sample generation, STaSy leverages probability flow ODEs to solve the inverse SDEs. Neural ODEs facilitate computing the log-probabilities of individual data records, enabling fine-tuning of the initially trained model, improving the diversity of generated minority class examples.

Building on the foundational concepts discussed, AutoDiffusion [32] presents an innovative solution for handling heterogeneous features in tabular data, combining the strengths of AEs and DDPMs or SGMs for data generation. This hybrid model leverages the capability of AEs to effectively deal with heterogeneous features and the proficiency of DMs in learning distributions in continuous space. In addition, this approach offers greater resource efficiency by performing the diffusion process within latent space, drawing parallels to Latent Diffusion Models (LDMs) [64] and demonstrating their applicability to synthetic tabular data generation tasks.

ForestDiffusion [31] takes a unique approach by combining SGMs with conditional flow matching. Unlike other techniques primarily relying on neural networks, it employs XGBoost [65], a renowned Gradient Boosting Tree (GBT) method. This approach duplicates the training dataset to compute the expectation in the diffusion and flow losses effectively. Instead of performing label conditioning within a single generative model, ForestDiffusion trains a separate XGBoost model for each class, forcing label-based splitting before deeper tree growth. This strategy enhances minority sample generation performance, as each model specializes in capturing the unique characteristics of a specific class regardless of the number of samples present in the data.

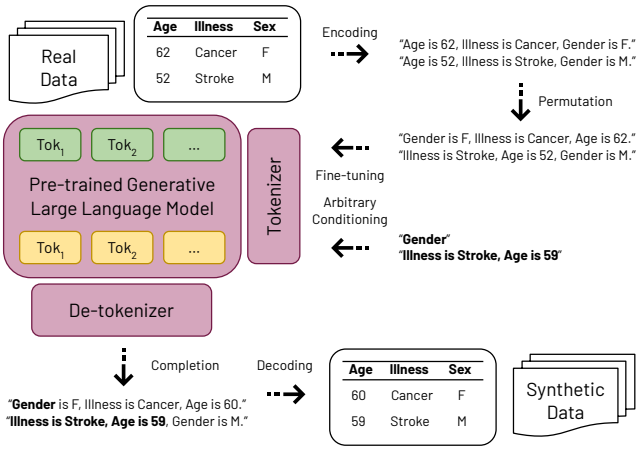

Figure 5: Depiction of the synthetic tabular data synthesis process using Large Language Models (LLMs). It begins with the conversion of tabular data into text, which is then permutated and tokenized as a preparatory step for fine-tuning the LLMs. For generating synthetic data, the models are conditioned with subsets of this text-based data, and the LLMs are tasked with generating the remaining features.

Although as mentioned above it is not a purely network-based approach, it nonetheless falls into the SGM category even if currently not using a neural network to estimate score functions.

For a comprehensive overview of DMs, readers are encouraged to consult [66]. Those specifically interested in the application of DMs to structured data can find an in-depth analysis in [67].

3.4 Large Language Models

LLMs are specific types of Natural Language Processing (NLP) models that utilize Transformers [68] and are characterized by their massive scale, often having billions of parameters. These models are trained on huge text datasets, achieving remarkable capabilities in language understanding and generation [69]. Prominent examples of LLMs include the GPT [70] family of models. Models like these define significant advancements in NLP, offering an ample range of applications due to their sophisticated understanding of language patterns and ability to solve complex tasks (via text generation). Their advanced text generation abilities open up potential avenues for novel applications, including the possibility of their use in this area [71].

LLMs follow similar architectural designs and pre-training objectives as smaller language models, but they significantly expand in three key areas: model size, data size, and computational power. A key advantage resulting from their massive scale is enhanced pattern recognition, enabling them to handle noisy or incomplete data. The vast knowledge encoded within LLMs presents a unique opportunity to generate highly diverse minority class samples that extend beyond the observed minority class manifold. This capability is particularly valuable in data-scarce scenarios, where LLMs can leverage prior knowledge to create synthetic samples that capture a wider range of potential variations within the minority class.

Scaling has been a critical factor in their development, as extensive research indicates that it can substantially enhance model

abilities [72]. The significant scaling up of LLMs has led to what is known in the literature as 'emergent abilities', a concept explored in [73]. Unique to LLMs, these abilities do not appear in smaller models and arise only at large scales. These emergent capabilities encompass in-context learning, meaning LLMs can understand and adapt to new information or tasks based on the given context; instruction following, demonstrating their capacity to comprehend and enforce complex instructions; and step-by-step reasoning, which allows them to process logically and reason through problems or queries methodically.

The application of LLMs to the problem of tabular data synthesis has been made possible through several developments primarily revolving around textual encoding (see Fig. 5). This process involves transforming tabular data into a text-based format [33], a critical step that enables fine-tuning LLMs for our purpose since they are inherently designed for processing and generating text. A significant technique facilitating this application is training models with textual encodings incorporating random feature order permutations. This approach allows the models to be arbitrarily conditioned. LLMs can model the data distribution conditioned on any selected group of features and then generate the remaining ones. This flexibility allows LLMs to understand and capture the complex nonlinear relationships and patterns present in tabular data.

GReaT [33], utilizes an auto-regressive LLM based on a transformer-decoder network architecture for sampling synthetic tabular data. The auto-regressive nature of GReaT ensures that synthetic data respects complex feature relationships, even when generating samples for minority classes. This is critical for maintaining the integrity of tabular datasets, where feature dependencies often play a significant role in downstream tasks. This approach involves converting tabular datasets into textual representations to leverage the capabilities of pre-trained self-attention-based LLMs. The conversion process is designed to tackle three key challenges. Firstly, it addresses the issue of lossy pre-processing, ensuring that crucial information from the tabular data is retained in the textual format (e.g., no artificial ordering is introduced in categorical variable conversion). Secondly, it focuses on maintaining coherent semantics, leveraging context knowledge inherent in LLMs to generate consistent data. Finally, it allows for arbitrary conditioning, meaning the model can generate data conditioned on any specific combination of features, which is particularly important for minority sample generation.

Tabula [34], builds upon the GReaT architecture. It addresses certain limitations inherent in using NLP models for this purpose, particularly long training and inference times. Tabula proposes a new foundational model specifically trained for tabular data synthesis. Leveraging a model that is already attuned to the particularities of tabular data potentially reduces the time and resources required for training and inference. Another key innovation in Tabula is a token sequence compression scheme designed to improve training times while preserving synthetic data quality, which is especially important for high-dimensional datasets. This scheme enhances the capture of interdependencies between tokens, which can benefit synthetic minority class sample generation. Additionally, Tabula introduces a new token padding method, which enhances the alignment of token sequences across the complete training batch, further optimizing the training process.

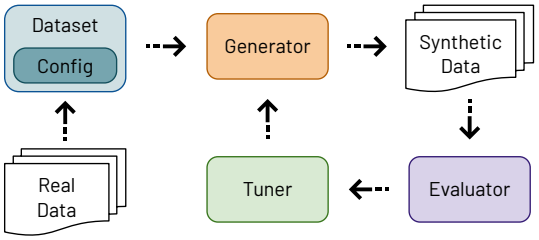

Figure 6: Workflow of the GenTab library. Real data is first input and pre-processed through the *Dataset* module, configured via a *Config* file. Next, a *Generator* is trained to produce synthetic data, which is then evaluated by the *Evaluator* module to compute performance metrics. These metrics guide the *Tuner* module in optimizing the generator's parameters to improve the quality of the synthetic data.

In contrast to previously discussed methods requiring fine-tuning of LLMs, often constrained by computational costs to older GPT models, alternative methodologies leverage pre-trained LLMs directly for tabular data generation. Notably, Curated LLM (CLLM) [74] harnesses the inherent knowledge of GPT-4 [75] without fine-tuning, offering a framework for tabular data generation particularly valuable in data-scarce scenarios like ours. CLLM employs a curated selection process, utilizing confidence and aleatoric uncertainty metrics derived from a supervised model trained on the available data to filter undesirable synthetic samples, thereby prioritizing high-quality data for downstream model training. To generate new data samples, the frozen LLM is provided with prompts containing background (text description of the dataset and task), examples (features and labels in text format), and instructions (the LLM is instructed to leverage the contextual information and provide generated samples respecting data relationships). This approach is particularly valuable in low-data regimes, where it can extrapolate to unseen regions of the data manifold based on contextual understanding of the features, effectively utilizing prior knowledge embedded in the pre-trained LLM. This enhanced coverage of the minority class manifold should improve diversity when generating new minority samples and enhance downstream model performance.

For those interested in a more detailed and in-depth exploration of LLMs, a comprehensive resource is available in [76]. A more specific survey centered on LLMs applied to tabular data is [77].

4 GENTAB: OUR SYNTHETIC TABULAR DATA GENERATION FRAMEWORK

To facilitate the use and reproducibility of synthetic tabular data generation methods, we have developed GenTab, an open-source library designed to simplify the process of generating synthetic data. GenTab provides a comprehensive set of tools for generating, tuning, and evaluating data using various state-of-the-art techniques. It was used in our own experimentation to benchmark and compare the methods discussed in this survey, ensuring that our results are reproducible. In this section, we provide an overview of GenTab's architecture, describing how it integrates different data generation models, its user interface, and the key functionalities it offers.

```
from gentab.generators import AutoDiffusion
from gentab.evaluators import LightGBM
from gentab.tuners import AutoDiffusionTuner
from gentab.data import Config, Dataset

config = Config("configs/adult.json")

dataset = Dataset(config)
dataset.merge_classes({
  "<=50K": ["<=50K."], ">50K": [">50K."]
})
dataset.reduce_mem()

trials = 10

generator = AutoDiffusion(dataset)
evaluator = LightGBM(generator)
tuner = AutoDiffusionTuner(evaluator, trials)
tuner.tune()
tuner.save_to_disk()
```

List. 1: Sample code to implement the GenTab workflow. In the code we parse a *Config*, create a *Dataset* and do some operations on it, create a *Generator*, create an *Evaluator*, create a *Tuner*, and run hyperparameter tuning for the desired generator finally storing the best dataset and model parameters that it has obtained after ten tries.

4.1 Overview

The library is structured around a well defined workflow to facilitate an intuitive and effective synthetic data generation process (see Fig. 6 and List. 1). It starts with a *Config* file, an element that holds information about the dataset, generation models, and downstream task, among other relevant parameters. Following the configuration setup, the *Dataset* module takes over. It uses the details from the configuration file to load and pre-process the data. This step is essential for converting the data to a format suitable for the different generative models. Once the data is ready, the *Generator* trains the chosen generative model with the pre-processed data and subsequently generates new synthetic samples. Once the data has been generated, the *Evaluator* assesses its quality. This module ensures the generated data meets predefined quality standards and is useful for its intended applications. The library also includes a *Tuner*, which employs hyperparameter tuning to optimize the performance of a specific generator method for a certain dataset.

4.2 Config

The configuration module utilizes JSON (JavaScript Object Notation) [78] to store information related to the library and datasets. The configuration file encompasses all the parameters required to tailor the synthetic data generation process to specific datasets and tasks. It includes the name and path to the dataset train and test splits. If desired, the user can specify to download common datasets present in either the Imbalanced Learn library [79] or the UCI Machine Learning Repository [80]. In addition, it stores other common dataset properties like categorical, binary, or integer column names. The user also needs to input the type of task (multi-class or binary classification) and the target label. In addition, some generative models require the user to input extra information related to column data distributions or other model-specific settings.

4.3 Dataset

This module is responsible for handling the dataset and loading it into memory, based on the configuration settings. After loading, it pre-processes the dataset to make it suitable for processing by the generators. Its functionalities cover a range of operations that are crucial for efficient data handling and preparation. These operations include writing the dataset back into a suitable format if the user desires to save it to disk, creating data frames that organize the data in a structured manner for the different generators, and reducing memory consumption, which is particularly important for handling large datasets. Additionally, this module also can perform random under-sampling of the dataset. This can be useful in scenarios where the dataset is imbalanced, with some classes significantly outnumbering others. By under-sampling, the module can create a more balanced dataset according to user preference. Our dataset module also includes tools for assessing the quality of synthetic datasets, focusing on fidelity and privacy.

Fidelity Regarding marginal fidelity, we use the Jensen Shannon divergence (JSD) [81], to help assess differences within categorical feature probability distributions. This metric is symmetrical and its interval is $[0, 1]$, which makes it well suited for comparisons. In a similar fashion, we use the Wasserstein distance (WD) [82] to estimate how well the distributions of continuous features are replicated, as it offers better numerical stability over JSD for this data type, as noted in [29]. The interval for WD is $[0, \infty]$, making it more difficult to compare across datasets. In terms of joint fidelity, we assess feature relationships and how well they are preserved in the synthetic data versus the real data using several correlation metrics. The Pearson Correlation Coefficient (PCC) [83] is utilized for pairs of continuous features, ranging in the interval $[-1, 1]$ and measuring the strength and direction of the relationship between those features. Next, the Theil's U (TU) Coefficient [84] is used to assess information dependencies between pairs of categorical features. Ranging in the interval $[0, 1]$, it quantifies the amount of information one feature reveals about the other. Additionally, the Correlation Ratio (CR) is utilized for measuring the dispersion of categorical and continuous feature pairs across the whole population. Its output value pertains to the $[0, 1]$ interval, zero meaning a category cannot be inferred from a continuous feature and one meaning the category can be obtained with total confidence. Finally, to derive a single comprehensive measurement, we calculate the ℓ^2-norms of all computed correlation measures within each dataset.

Privacy We utilize Distance to Closest Record (DCR) [29], specifically computing the minimum ℓ^2-norm from the synthetic samples to the real records, and averaging these distances across all synthetic data to yield the Mean-DCR (MDCR). Since its result belongs to the $[0, \infty]$ interval, this metric is difficult to compare across datasets. A high MDCR indicates greater privacy, while a low MDCR raises privacy concerns. However, random noise can produce artificially high DCR values, so high DCR alone does not guarantee privacy. It is crucial to consider fidelity and utility metrics alongside DCR, as noted in [32]. We also include the Nearest Neighbor Distance Ratio (NNDR) [85], which compares distances between consecutive nearest neighbors of synthetic and real samples. This metric produces a ratio

in the [0, 1] interval. Higher values indicate better privacy, as they suggest less proximity to sparse outliers in the original dataset. In contrast, an NNDR near zero indicates synthetic data points close to original points in sparse regions, while NNDR values approaching one suggest synthetic samples are located within dense regions of the original data. Additionally, we provide the Hitting Rate (HR) [86], a membership inference metric that identifies close matches (within a chosen threshold, 3 % in our case) between synthetic and real data. This metric ranges within the [0, 1] interval. Lower values are preferable, as they indicate a lower ratio of closely replicated real samples in the synthetic data. Lastly, Epsilon Identifiability Risk (EIR) [87] measures the proportion of real data points that have a generated sample closer than the next-nearest real data point. Distances are weighted by the inverse of each column's entropy to emphasize rare data points. This metric also produces values in the [0, 1] interval, with lower values meaning better privacy.

4.4 *Generator*

The Generator module is the foundation for all the generative models implemented within the system. This module is tasked with essential functions like pre-processing (if needed), training, and sampling synthetic data. Its design is user-friendly, facilitating the integration of new models with ease. To incorporate a new generative model into the library, users simply need to create a child class derived from the *Generator* base class. Within this child class, they are required to implement three key functions: *pre-process* for any model-specific data pre-processing, *train* for training the generative model using the pre-processed data, and *sample* for generating new synthetic data based on the trained model. Reference implementations for the generative models presented in Table 2 are organized within this module. Each model implementation resides in its own source file and follows the implementation guidelines outlined above. This structured approach not only enhances the usability of the library but also provides a framework to explore, implement, and compare different generative models.

4.5 *Evaluator*

The evaluation module is a critical component that offers a range of evaluators for testing the quality of synthetic datasets for the chosen downstream task. The evaluators interact with the tuning module to provide key metrics on dataset performance in ML tasks. These metrics are essential for the tuning module to tailor the data generation model to each specific dataset, ensuring it accurately replicates its underlying characteristics. This module also allows for the integration of custom evaluators, enhancing its versatility. Users can leverage the *Evaluator* base class and create a child class that implements the necessary evaluation functions: *pre-process* for any evaluator-specific data pre-processing, *fit* for training the evaluator model, *predict* for making predictions with the trained model, and *post-process* for any needed post-processing of the evaluator results.

The library currently includes implementations of several state-of-the-art evaluators. We selected LightGBM [88], a gradient boosting library that has tree based learning algorithms at its core. Additionally, we included XGBoost [65], which, although more computationally expensive, can yield better results. CatBoost [89], another gradient boosting framework, was chosen for implementing a different approach for processing categorical features, a permutation driven algorithm. To further diversify our evaluation methods, we incorporated a classical technique, a linear Support Vector Machine (SVM) [90]. Finally, we implemented a Multilayer Perceptron (MLP) [91] specially tailored for tabular data classification tasks. This architecture yielded the fastest and simplest model with competitive performance among neural network-based evaluation methods for the problem at hand.

ML Utility The Evaluator module automatically computes standard metrics to evaluate the quality of the generated dataset. These metrics include the Matthews Correlation Coeficient (MCC) [92], which measures classification quality accounting for true and false positives, with values ranging from -1 (perfect inverse prediction) to 1 (perfect prediction), and 0 meaning a random prediction. Additionally, it computes accuracy (overall correctness of predictions), precision (exactness of predictions for a specific class), recall (percentage of total occurrences of a class that the model can accurately detect), and the F-Score (harmonic mean of precision and recall). To address class imbalance, the module provides simple (macro) and weighted averages for suitable metrics. This ensures an accurate evaluation of the synthetic data, particularly in our task at hand, where some classes are underrepresented. All of these metrics range in the interval [0%, 100%].

4.6 *Tuner*

Since hyperparameter optimization is a key step in any neural network approach, we provide the *Tuner* module, which allows the user to perform hyperparameter tuning for any *Generator* in the library. For this task we have chosen Optuna [93], a hyperparameter tuning library that employs advanced methods in its parameter sampling and pruning mechanisms. Optuna uses techniques like the Tree-structured Parzen Estimator (TPE) [94] for sampling, which creates a probabilistic model based on past trials to suggest new promising combinations. It also implements an alternative, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [95], a robust black-box optimization method. For pruning, it uses strategies like the Asynchronous Successive Halving Algorithm (ASHA), an extension of [96] to stop less promising trials early, saving resources and improving efficiency in finding optimal solutions.

We provide several default hyperparameter combinations to simplify the tuning process, but users can also customize them as needed. These default settings are based on combinations provided by the original authors and have demonstrated strong performance in our testing. As with the other modules involved in the data generation process, we provide reference implementations and facilitate the tuning of new generative models. Users just need to create a new child class of the *Tuner* base class that implements the *objective* function, which instantiates a new *Generator* and evaluates it using the chosen *Evaluator*.

5 RESULTS

This section presents the experimental results obtained from evaluating the synthetic data generators implemented in our library. As commented, these generators were selected as representative

Table 3: Dataset characteristics, including imbalance ratio, training and testing splits, and the number of continuous and categorical features.

Dataset	Imbalance Ratio	Train/Test Split	Cont.	Cat.
Car Evaluation [24]	18.62	1.12K / 0.61K	0	6
PlayNet [97]	3.17	73.60K / 301.02K	66	0
Adult [98]	3.18	39.07K / 9.77K	6	8
Ecoli [99]	8.57	0.22K / 0.12K	0	7
Sick Euthyroid [100]	9.78	2.06K / 1.11K	0	42
Calif. Housing [101]	5.49	16.51K / 4.13K	8	0
Mushroom [102]	2.69	4.74K / 1.62K	0	22
Oil [99]	21.48	0.61K / 0.33K	0	49

examples based on the methods discussed in Section 3 of this survey. The evaluation encompasses four dimensions: computational efficiency, fidelity, privacy, and ML utility. Computational efficiency provides insight into the resource requirements of each method, while fidelity and ML utility assess the suitability of the synthetic data as a substitute for the real dataset in practical applications. Privacy, on the other hand, measures the risk of sensitive information being exposed if the synthetic data is shared or leaked. The evaluation aims to identify the strengths, weaknesses, and computational trade-offs of the different methods, with particular emphasis on their effectiveness in addressing class imbalance.

We performed all tests on an AMD EPYC 7763 with 256 GB of RAM, a 2TB SSD, a single NVIDIA A100 80GB GPU, and the Ubuntu Linux 18.04 x64 operating system. To deal with the inherent randomness in the data generation process, we use fixed seeds for both generator and evaluator models for reproducibility purposes. Additionally, we generate multiple synthetic datasets (the same number for each generative model) and select the best-performing one based on the chosen evaluation metrics.

5.1 Datasets

The chosen datasets for our study cover a wide range of characteristics, such as class distribution, feature types, and dataset size, to effectively test synthetic data generation methods. Together, these datasets offer a comprehensive basis for evaluating the performance and versatility of synthetic tabular data generation methods.

Car Evaluation We utilized a used car evaluation dataset from [24]. This dataset is comprised of entirely categorical features. To specifically test highly imbalanced scenarios, we have used the version found in [99], which has been binarized. This particular version of the dataset has been pre-processed to represent a binary classification problem with a markedly imbalanced class distribution. The target task in this dataset is predicting the second hand car evaluation quality based on car attributes like price, tech or comfort. It represents scenarios where decisions are made based on qualitative assessments rather than quantitative measurements.

PlayNet We also included a handball play classification dataset, detailed in [97], due to its high dimensionality (66 features) and predominantly numeric features. This large dataset, resulting from high-frequency sampling during the handball matches, comprises over one million rows. In our study, we addressed the challenge of handling a large dataset by selecting a subset of approximately ~74K randomly chosen samples. This approach was necessary due to time and performance constraints, but we ensured that the subset maintained a similar imbalance ratio and achieved competitive performance on the full test dataset. It includes player positions and velocities obtained in two distinct handball arenas and across several different games. These features hold geometric and physical significance, enabling a deeper visual analysis of the generated synthetic datasets. The target task for this dataset is to classify the type of game situation [6] in progress at specific game times using player and ball dynamics. This dataset offers a complex scenario typical in sports analytics and automatic production.

Adult We selected the UCI Adult dataset, as referenced in [98], which provides a mix of categorical, binary, and numerical features with intermediate dimensionality. This dataset is commonly used in machine learning research and presents a balanced blend of feature types. The dataset comprises approximately 50K records, each containing 14 features. The target task is predicting whether income is higher than $50K. This dataset offers a scenario typical in financial and social science contexts.

Ecoli The biology dataset [103], containing protein localization site data, and more specifically, its binarized version [99], was also included in our analysis. The target task is predicting the cellular localization sites of proteins. This dataset exhibits significant class imbalance and presents the fewest samples among our chosen datasets, allowing us to assess how methods perform with very scarce data.

Sick Euthyroid This medical dataset [100] focuses on thyroid disease classification. It includes patient attributes to predict the presence of hypothyroidism, a task complicated by significant class imbalance typical in medical domains. This dataset allows us to evaluate how effectively these data generation techniques mitigate imbalances in the healthcare domain.

California Housing This housing dataset [101], derived from the 1990 California census data, incorporates one sample per census block. A block is the smallest unit the U.S. Census Bureau employs to publish sample data. The target variable is the median house value for California, which is noted in hundreds of thousands of dollars. Median house value for California districts, was quantized into distinct price groups, creating a classification problem. The spatial structure of this dataset, containing latitude and longitude coordinates for each sample, provides an ideal opportunity to assess the ability of each model to preserve geographic relationships.

Mushroom [102] describes 23 gilled mushroom species (Agaricus and Lepiota family) and classifies them as edible or poisonous. Categorical features represent distinct mushroom characteristics used to predict edibility. The target variable is mushroom edibility. This dataset represents another biological dataset, but in this case comprised of descriptive categorical features. It also provides a intermediate amount of features and samples, and a low imbalance ratio.

Oil This environmental dataset, [99], our most imbalanced, originates from satellite images categorized as containing oil

Table 4: Timing for the data fitting (hours) and sampling processes (seconds), with default and tuned model parameters for the **Adult** dataset. We can see that AEs are the fastest training and sampling network-based methods, while DMs are the slowest in training and LLMs in sampling. **Bold** highlights fastest network-based methods and underlined highlights the slowest.

Model	Baseline		Tuned	
	Fit (h)	Sample (s)	Fit (h)	Sample (s)
SMOTE [13]	-	46.18	-	83.36
ADASYN [14]	-	0.31	-	3.56
TVAE [28]	**0.03**	**1.50**	**0.11**	6.13
CTGAN [28]	0.04	2.59	0.42	**2.63**
GaussianCopula [16]	-	0.84	-	0.86
CopulaGAN [28]	**0.03**	3.72	**0.11**	3.95
CTAB-GAN [29]	0.18	6.10	0.66	6.20
CTAB-GAN+ [30]	0.21	7.05	0.66	7.05
AutoDiffusion [32]	0.83	1.71	17.60	3.06
ForestDiffusion [31]	13.14	114.53	3.84	12.17
GReaT [33]	0.74	631.25	1.62	773.98
Tabula [34]	0.63	584.89	0.94	274.93

spills or not. Image sections were processed to extract descriptive feature vectors. The task is to classify these patches as an oil spill or non-spill, reflecting the real-world challenge of detecting environmentally damaging oil spills in the environmental domain.

For a more detailed description of the datasets, like imbalance ratios, train-test splits, or number and type of features, please refer to Table 3. We approximately use 5 % of the training splits for model validation.

5.2 *Computational Efficiency*

Identifying the lack of emphasis on computational requirements within the existing literature, we evaluate both training (fit) and generation (sample) execution times across the 12 generators implemented in our library. Our tests measure training and generation time using both default and tuned hyperparameters, as parameter choice significantly impacts the computational cost of each method. This approach allows us to see the difference between default and optimal generators for a certain evaluator in terms of execution times. We chose the Adult dataset to perform this test due to time and computational constraints, and its balanced representation of factors (categorical and numerical features). We concentrated our efforts on dataset balancing performance since it is the focus of our survey. Table 4 displays the execution times when the generators were tasked with creating 19K new samples for balancing the minority class.

The methods that performed best were TVAE and simpler GAN approaches. AEs offer lower training costs as they lack computationally expensive transformers or diffusion processes. CopulaGAN did well due to internally using GaussianCopula, which, depending on the selected statistical estimator, can be more computationally efficient. While default parameters in these methods generally strike a balance between model fidelity and training cost, ForestDiffusion stands as an exception. Its default configuration proved too computationally expensive for timely execution. The baseline ForestDiffusion results presented in Table 4 were obtained using a parameter configuration adjusted to prioritize computational efficiency.

Table 5: Evaluation of synthetic dataset fidelity using a combination of statistical measures. We provide the average rank for Jensen Shannon divergences (JSD) and Wasserstein distances (WD) assessing the marginal fidelity of categorical and continuous features. Additionally, we show ℓ^2-norms for Pearson Correlation Coefficients (PCC), Theil's U (TU), and Correlation Ratios (CR) to evaluate the preservation of feature relationships within the datasets. Lower is better. Best results are highlighted in **bold**, second best are underlined.

Model	Marginal		Joint		
	JSD	WD	PCC	TU	CR
SMOTE [13]	4.25	5.33	**0.42**	2.67	**0.04**
ADASYN [14]	5.50	**2.00**	0.48	2.99	0.12
TVAE [28]	**3.75**	9.00	1.97	5.63	0.08
CTGAN [28]	6.00	5.00	2.50	2.91	0.17
GaussianCopula [16]	5.25	8.33	1.15	2.99	0.18
CopulaGAN [28]	10.25	5.00	2.49	2.88	0.16
CTAB-GAN [29]	7.50	5.33	2.20	2.47	0.09
CTAB-GAN+ [30]	6.75	7.67	2.38	1.21	0.05
AutoDiffusion [32]	6.75	8.67	2.85	2.73	**0.04**
ForestDiffusion [31]	10.50	**2.00**	0.45	1.50	0.17
GReaT [33]	4.50	10.33	1.53	2.25	**0.04**
Tabula [34]	7.00	9.33	1.64	**0.25**	0.06

DMs such as ForestDiffusion, contingent on parameter selection, can present training challenges on a single machine, not only in terms of time, but also disk space when training in parallel. Depending on the selected hyperparameters it can be the slowest method by a wide margin, sometimes even taking weeks to train in our test machine. AutoDiffusion offers acceptable training and sampling times. However, we have seen ML utility improvements when making its architecture deeper, which leads to longer training times, the reason why when tuned, it became slower than ForestDiffusion. Conversely, in this test, ForestDiffusion performed better when reducing the amount of dataset replication, estimators, and tree depth, leading to faster training times and better performance. LLMs, specifically GReaT and Tabula, exhibit slow sampling due to model size, increasing generation times for highly unbalanced datasets.

As anticipated, certain statistical and local methods demonstrate greater computational efficiency than network-based approaches due to having less underlying complexity. Thus, users with hardware or time constraints may initially favor these models. While in some instances they may yield sub-optimal results, they offer rapid training and evaluation.

5.3 *Fidelity*

Our evaluation includes both marginal distribution metrics (JSD and WD) and joint distribution metrics (PCC, TU, and CR). To facilitate comparisons across diverse datasets, we report the average ranking of each method over all tested datasets for marginal metrics, since distances across datasets have a high amount of variance and thus can corrupt results across datasets. For joint distribution metrics, since correlations share the same output ranges, we directly average the results rather than the rank. All models were trained using the corresponding training split for

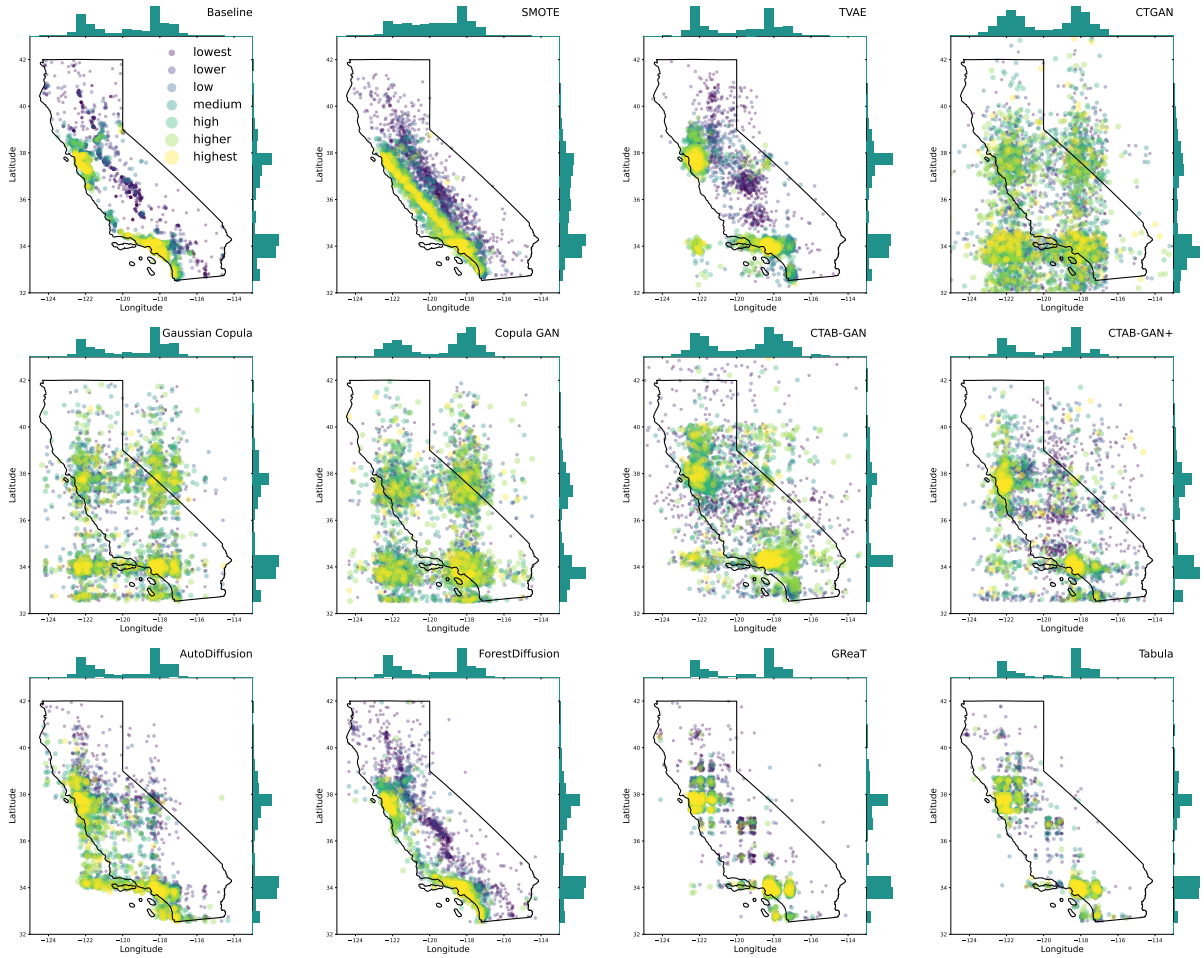

Figure 7: Comparison of real and synthetic samples for the **California Housing** dataset. House value class is indicated by point color and size, while joint histograms of latitude and longitude illustrate the variable's spatial distribution. The California state boundary is demarcated by the black outline. This dataset reveals limitations in most methods except Forest Diffusion, with TVAE and AutoDiffusion also producing reasonable, though less accurate results. LLMs showed middle ground performance, while GANs struggled to reproduce the dataset distributions.

each real dataset. We present metrics and visualizations for the fully synthetic data generated by the different methods. Table 5 shows how the tested generative models performed across the selected datasets and metrics.

ForestDiffusion excels in numerical feature modeling, achieving high WD and PCC scores, significantly outperforming other network-based methods, reflecting the strength of DMs when modeling continuous data. TVAE performed exceptionally well when preserving categorical variable distributions, obtaining the best JSD score. When it comes to categorical feature relationships, Tabula and ForestDiffusion performed best. We attribute the success of Tabula to the NLP abilities of LLMs, obtaining the best TU score. Its leading performance in the TU metric demonstrates its effectiveness when maintaining categorical variable relationships. Lastly, GReaT and AutoDiffusion exhibit strong performance when modeling mixed categorical-continuous correlations with the highest CRs, outperforming other generative models and showcasing their ability to capture complex feature dependencies. Nearest neighbor-based methods performed above average in quantitative metrics, but this performance is achieved at the expense of privacy (see Table 6) and data distribution distortion.

Additionally, to complement the quantitative analysis presented in Table 5, we performed a qualitative study on two datasets that allow better visual interpretation of the generated synthetic data. Thus, Fig. 7 presents a scatterplot visualizing the spatial distribution of median house value classes for the synthetic datasets in the California Housing dataset. Geographic information proves challenging for most generative models, particularly GANs, which tend to produce grid-like patterns with numerous invalid points extending beyond state boundaries or the sea. Ideally, generated synthetic data should not only exhibit realistic distributions for house values, latitude, and longitude independently but also preserve the spatial correlations between these variables. For instance, synthetic data points should realistically reflect the distribution of house values across different geographical locations within California. Methods that fail to capture these spatial relationships may generate implausible data points, such as houses located in the ocean or outside of California altogether. While these models correctly identified individual

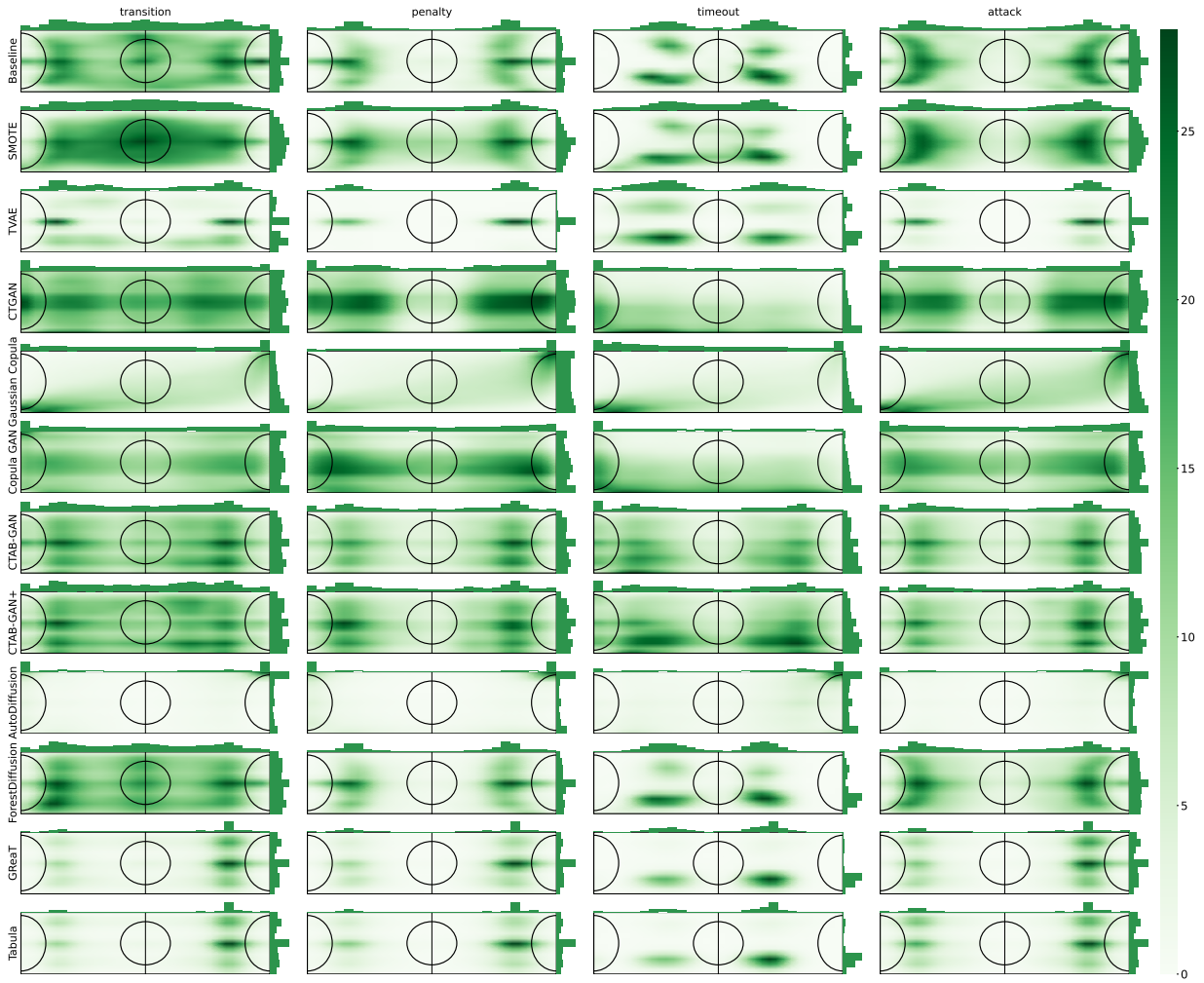

Figure 8: Player position density histograms and heatmaps for the synthetic data versus the original data in the **PlayNet** dataset. Darker colors in the heatmap represent a higher density of player positions within that specific area of the court, signifying an increased likelihood of finding players in those regions throughout a game situation. We can clearly see which methods preserve player behaviour patters (ForestDiffusion, CTAB-GAN+, and to some extent Tabula and GReaT), and which do not (TVAE, CTGAN, GaussianCopula, AutoDiffusion). These patterns are specially noticeable in the timeout and penalty game situations.

variable distributions in some cases, they did not capture global relationships accurately, leading to incorrect final results. LLMs also generate plausible individual coordinate histograms, but their spatial distributions and densities are inaccurate, albeit with fewer out-of-bounds points than GANs. TVAE and AutoDiffusion show improved performance, reasonably reproducing sample distribution and density, but still generating some geographically nonsensical points (e.g., located in the sea). In contrast, ForestDiffusion excels, uniquely capturing the underlying geographic relationships between latitude, longitude, and median house value, with accurate variable distributions, class densities, and minimal out-of-bounds points. While randomized and probabilistic methods exhibited strong performance in some quantitative metrics, visual assessments reveal distortions in data distributions, highlighting the risk of reaching misleading conclusions when relying solely on numerical evaluations.

Lastly, in Fig. 8, we present player position density histograms and heatmaps, generated using kernel density estimation (KDE) [104], for all game situations (timeout, transition, attack, and penalty) in both real and synthetic PlayNet datasets, illustrating player spatial distributions across game states. Most methods struggled to capture player positioning during transitions. CTAB-GAN and CTAB-GAN+ achieved better results, while others showed deviations from the real data distribution, depicting players either scattered across the court with no apparent governing patterns or concentrated in a few key areas, with only ForestDiffusion accurately modeling positioning density in this game situation. Similarly, ForestDiffusion uniquely captures player positioning during penalties, with all other methods except LLMs failing to provide correct player densities. For timeouts, TVAE, ForestDiffusion, GReaT, and Tabula effectively replicate player density patterns. In attack situations, ForestDiffusion and the CTAB-GAN variants reasonably reproduce underlying positioning patterns. Randomized and probabilistic methods exhibit similar problems to the previous test case, distorting player densities and movement patterns across game situations. Single samples for each method and game situation

Table 6: Evaluation of privacy properties of synthetic data generation methods across all tested datasets. We include Mean Distance to Closest Record (MDCR) rankings and average Nearest Neighbor Distance Ratio (NNDR). Higher is better. Additionally, we provide average Hit Rate (HR) and average Epsilon Identifiability Risk (EIR). Lower is better. Best results are highlighted in **bold**, second best are underlined.

Model	MDCR Rank	NNDR	HR	EIR
SMOTE [13]	3.83	0.66	0.14	0.38
ADASYN [14]	3.83	0.52	0.19	0.45
TVAE [28]	4.00	0.79	0.12	0.18
CTGAN [28]	7.50	**0.82**	**0.09**	0.15
GaussianCopula [16]	8.17	0.81	0.11	**0.12**
CopulaGAN [28]	**8.83**	0.81	**0.09**	0.15
CTAB-GAN [29]	8.33	0.78	0.11	0.21
CTAB-GAN+ [30]	7.67	0.79	0.11	0.24
AutoDiffusion [32]	8.50	0.76	0.16	0.29
ForestDiffusion [31]	6.83	0.80	0.19	0.28
GReaT [33]	5.67	0.66	0.20	0.32
Tabula [34]	4.83	0.35	0.46	0.45

are provided in the Appendix.

Cross-correlation distance heatmaps to better ascertain visually how the different methods stack up in terms of fidelity for each individual dataset are given in the Appendix.

5.4 *Privacy*

To assess privacy, we first average MDCR ranks across all datasets. Focusing on ranks mitigates dataset-specific variability, enabling a robust comparison of the privacy preservation capabilities of the selected methods. We also provide average NNDR values, a metric that offers information about distance between synthetic samples and their two closest real samples, complementing MDCR. Additionally, we use the average HR metric to detect sample replication. Finally, average EIR estimates the proportion of original samples with a synthetic neighbor closer than the nearest real neighbor. All models were trained on the respective training split for each real dataset. Subsequently, results were generated using the fully synthetic data produced by these trained models. Table 6 provides a detailed comparison of how the selected approaches fare regarding all the chosen metrics.

Among these approaches, CTGAN and CopulaGAN excel as top performers, consistently ranking among the best or second-best across all metrics. Gaussian Copula also performs well, as its statistical nature, while potentially limiting fidelity and global relationship preservation, enhances privacy. AutoDiffusion emerges as a strong contender among DMs, demonstrating good privacy preservation results, ranking second in MDCR, achieving good NNDR values, and exhibiting low HR and EIR. ForestDiffusion shows less impressive results, with slightly above-average performance in several metrics. TVAE presents mixed results regarding privacy. While its NNDR is reasonable, its below-average MDCR raises concerns. However, its above-average HR and EIR values suggest moderate privacy preservation capabilities. LLMs show below-average MDCR, NNDR, HR, and EIR values, suggesting poor privacy preservation capabilities in their current form. SMOTE and ADASYN are among the worst performers in most metrics due to their nearest-neighbor nature, which limits diversity and tends to closely imitate original dataset samples.

5.5 *ML Utility*

To assess the ML utility preserved in the generated datasets, we adopted the Train on Synthetic and Test on Real (TSTR) [105] approach. This methodology divides each dataset into training and testing splits. Next, after the generation of synthetic data based on the real training splits, we leverage the implemented evaluators to test the generated synthetic datasets with real testing data. Consequently, these evaluators are trained using their corresponding synthetically oversampled versions. We utilized the averages of five widely recognized metrics for each evaluator to provide comprehensive insight into ML method performance on the selected datasets: MCC, Accuracy, precision, recall, and F-Score. To better address class imbalance issues, we calculated macro and weighted averages for suitable metrics. Additionally, to facilitate comparison across the numerous metrics and methods, we provide the average rank for each method based on the most relevant metrics for imbalanced data (MCC and macro averages for precision, recall, and F-Score), as shown in Table 7.

The main conclusion we can draw from these results, is that DMs possess the most ML utility and provide the best or second-best results in almost all metrics, beating even the original dataset in the majority of imbalanced metrics. In terms of the most relevant results, ForestDiffusion shows an improvement of 0.06 in MCC, ~2.2 % in macro precision, ~4.9 % in macro recall, and ~2.9 % in macro F-Score. AutoDiffusion also beats the original dataset in terms of MCC by 0.05, macro precision by ~0.2 %, macro recall by ~5.3 %, and macro F-Score by ~2.9 %. TVAE also presents good results, improving on the original dataset in MCC but by a lesser margin 0.04, improving in terms of macro recall by ~4.5 % and macro F-Score by ~1.6 %. Among GAN methods, the best contender is CTAB-GAN, also improving on the original dataset MCC by 0.03 and on macro recall by a ~5.4 % margin. LLMs demonstrate poor performance, failing to improve upon the original dataset in any of the imbalanced metrics. SMOTE and ADASYN perform well on average across most metrics, but it is important to note that they did not obtain the best results on a per-dataset basis.

Detailed ML utility metrics and further commentary to better understand how each model behaves with each individual dataset are given in the Appendix.

5.6 *Discussion*

The results from our study offer several insights into the use of network-based generative models for synthetic tabular data generation in imbalanced scenarios. The following paragraphs outline the principal conclusions from our experimental analysis.

Computational Efficiency An overlooked topic in most of the literature are the computational requirements for tabular synthetic data generation methods. As we can see in the performance results, some of them have excellent ML utility and fidelity at the cost of high computational and time requirements. ForestDiffusion illustrates this point well. While consistently

Table 7: Average rank, MCC, accuracy, precision, recall, and F-Score for the tested methods and datasets. Best results are highlighted in **bold**, second best are underlined.

Model	Rank	MCC	Acc.	Precision		Recall		F-Score	
				Weighted	Macro	Weighted	Macro	Weighted	Macro
None		0.61	**89.2**%	88.0%	78.0%	**89.2**%	76.0%	**88.2**%	76.0%
SMOTE [13]	3.2	0.65	86.3%	88.9%	77.4%	86.3%	**82.0**%	87.0%	77.8%
ADASYN [14]	4.0	0.66	85.7%	89.0%	77.3%	85.7%	81.8%	86.4%	77.2%
TVAE [28]	5.0	0.65	86.8%	88.6%	77.5%	86.8%	80.5%	87.2%	77.6%
CTGAN [28]	8.2	0.63	86.2%	88.5%	76.1%	86.2%	79.8%	86.5%	75.9%
GaussianCopula [16]	9.2	0.62	87.3%	88.3%	77.1%	87.3%	78.4%	87.1%	75.7%
CopulaGAN [28]	7.5	0.63	86.3%	88.4%	76.9%	86.3%	79.6%	86.7%	76.2%
CTAB-GAN [29]	7.0	0.64	83.9%	87.7%	75.5%	83.9%	81.4%	84.6%	75.9%
CTAB-GAN+ [30]	9.0	0.62	83.0%	87.5%	73.8%	83.0%	81.2%	84.0%	74.4%
AutoDiffusion [32]	**2.2**	0.66	87.9%	88.8%	78.2%	87.9%	81.3%	88.1%	**78.9**%
ForestDiffusion [31]	2.5	**0.67**	87.5%	**89.2**%	**80.2**%	87.5%	80.9%	87.7%	78.9%
GReaT [33]	13.0	0.45	71.7%	84.3%	63.8%	71.7%	73.7%	75.5%	62.2%
Tabula [34]	12.0	0.50	77.4%	83.5%	66.0%	77.4%	74.6%	79.6%	66.2%

outperforming other methods, it does so at a significant computational cost. Training with default parameters took a long time and in terms of parallel training it required a high amount of disk space. The tuning of this model for certain datasets proved complicated due to its high computational cost. In general, DMs and LLMs proved costly to train and tune, whereas AutoDiffusion, with its latent space approach to the diffusion problem, proved less computationally expensive with certain configurations. GANs occupied a middle ground in terms of training cost, making them attractive for constrained resource environments, due to their good results in privacy, fidelity, and ML utility tests. Lastly, AEs generally demonstrated the lowest training and tuning costs when choosing reasonable parameter configurations. Local methods, like SMOTE or ADASYN, have proved to be the cheapest option. However, their cheapness comes at a cost: not preserving global relationships, data distribution distortion, and privacy, three important factors.

Fidelity Regarding fidelity, DMs unequivocally have proven superior performance. They excel in most marginal and joint quantitative metrics and qualitative assessments. ForestDiffusion consistently exhibits above-average performance across all tests, demonstrating exceptional capacity for modeling continuous variables and preserving global relationships within datasets. While AutoDiffusion emerges as the leader in continuous-categorical relationship preservation, its qualitative performance is less robust, particularly in one of the chosen visual assessments, the California Housing scatterplot. GANs have proven acceptable quantitative performance, with CTAB-GAN+ standing out for its consistently strong results across all tested metrics, albeit not being top-ranked. LLMs perform well quantitatively, securing top scores in categorical feature relationship preservation and achieving above-average results in all metrics except WD, but their qualitative performance is limited, often producing unrealistic samples with noticeable artifacts. TVAE, representative of the AE architecture, exhibits middle-ground performance in most metrics, but presents great results in categorical fidelity measures and feature correlation preservation. Randomized methods obtain results that at first glance seem good, performing well above average. However, closer inspection reveals that this performance is achieved at the expense of privacy and diversity, as these methods fail to generate samples outside the convex hull of neighboring points and tend to distort local distributions. Conversely, probabilistic methods perform well when modeling local numerical feature distributions but show subpar performance in other metrics, achieving poor results regarding feature correlation preservation.

Privacy GAN-based approaches, particularly CTGAN and CopulaGAN, emerge as top performers. This strong performance may be attributed to these methods incorporating techniques to model marginal distributions (e.g., via copulas), inherently adding a layer of abstraction from individual data points. TVAE achieved moderately good results in most privacy metrics except MDCR, which exposes the inherent weakness of AE architectures regarding privacy, memorization of records. AE-based methods are prone to overfitting, potentially leading to close reproduction of dataset records and compromising privacy. DMs provide a reasonable middle ground, with better MDCR values, but on average close to AEs in terms of privacy preservation. LLMs, however, demonstrate weaker performance than DMs, with both tested models exhibiting below-average results. A tailored fine-tuning process with a focus on privacy might have improved these results, but such an investigation falls outside the scope of this survey, which primarily focuses on addressing class imbalance. Randomized methods lacking explicit privacy-enhancing techniques perform poorly, as evidenced by the SMOTE and ADASYN results, obtaining most of the worst results among the evaluated approaches. However, probabilistic approaches, such as GaussianCopula, fare relatively well regarding baseline privacy. This is likely because they model marginal distributions separately and capture dependencies explicitly. This modeling process inherently focuses on aggregate properties and statistical relationships rather than preserving fine-grained details of individual records.

It is important to note that for serious privacy requirements involving sensitive data, dedicated techniques like Differential Privacy [106] must be integrated into the training or generation process of these methods. Our testing only reflects baseline tendencies without advanced privacy guarantees.

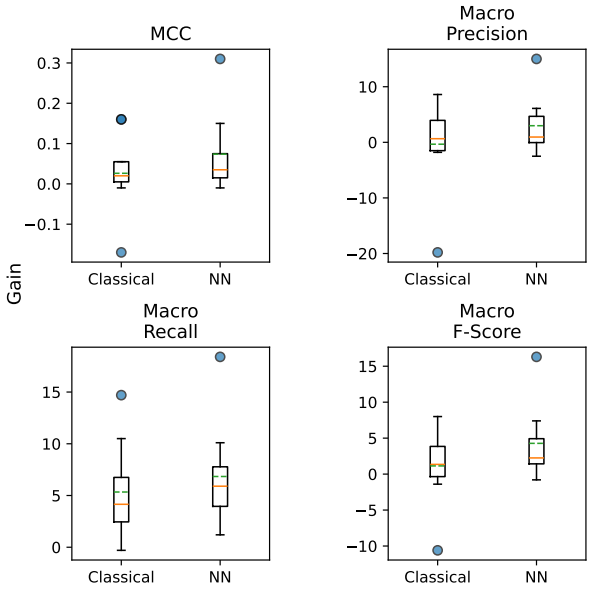

Figure 9: Box plots of all imbalanced metrics comparing the performance of neural network-based approaches to classical methods with key statistical measures highlighted for clarity. These include the mean (slashed green line), median (orange line), quartiles, and outliers. A positive skew is evident in all imbalanced metrics for neural network-based methods, suggesting better ML performance in imbalanced datasets versus traditional methods.

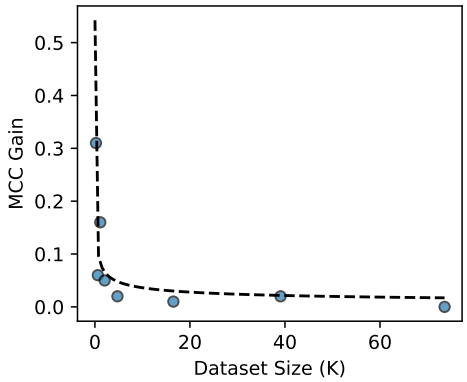

Figure 10: Chart displaying the effectiveness of network-based oversampling techniques across varying dataset sizes. A fitted logarithmic curve highlights the underlying trend: as dataset size decreases, the performance of synthetic data generation techniques improves.

ML utility When considering utility in our downstream task, DMs again demonstrate exceptional performance, surpassing all other methods and even the original datasets in most imbalanced classification metrics and some general metrics like weighted precision. While sometimes they fall short of SMOTE and ADASYN in macro recall (expected due to the nearest-neighbor nature of these methods), they outperform all other techniques and the original baselines. TVAE, representing purely AE-based architectures, emerges as the second-best performing family of models, outperforming the original dataset in MCC, macro recall, and macro F-score. GAN methods, led by CTAB-GAN, secure the next best results, with CTAB-GAN surpassing the original dataset in terms of MCC, macro recall, and macro F-Score. However, other GAN methods show less favorable results, although remaining competitive with the original dataset. LLMs obtained the worst results, struggling to improve in any imbalanced classification metric, trailing the original dataset in machine learning utility by a significant margin. This suggests that the tested LLMs, in their current form, may not be well-suited for oversampling and mitigating class imbalance in downstream tasks.

Fig. 9 provides box plots comparing the performance of classical and neural network-based methods. Neural network-based methods present a positive skew, with superior median, average, and positive outlier outcomes across all imbalanced metrics, indicating better overall performance. Neural network-based methods also exhibit better minimum results, suggesting greater robustness and less susceptibility to overfitting. These results highlight the ability of neural network-based methods to effectively capture and represent the underlying structure of minority classes while maintaining consistency with the overall data distribution, leading to more reliable performance in imbalanced scenarios.

Results suggest that, in general, datasets that benefit the most from synthetic data generation are the smallest ones, which resulted in the highest ML utility gains. Fig. 10 depicts the relationship between dataset size and the performance gains achieved. It presents a scatter plot comparing the size of the original datasets against the improvement in MCC attained by the best-performing network-based synthetic data generation methods. The plot illustrates a clear trend: smaller datasets exhibit substantial gains in MCC, with improvements reaching up to 0.3 in certain cases. As dataset size increases, the impact of synthetic data generation on MCC becomes less pronounced, eventually plateauing as the original dataset size becomes sufficiently large. This observation aligns well with our expectations, the addition of synthetic samples to small datasets can effectively augment training data, providing downstream models with a more comprehensive representation of the underlying data distribution, leading to improved generalization and enhanced performance on the target task.

6 CONCLUSIONS

Synthetic tabular data generation has proven as a valuable alternative for mitigating class imbalance issues. The application of neural network-based approaches in this domain presents distinct advantages over more traditional local methods. One key advantage of neural networks is their ability to account for the underlying data distribution, which leads to the generation of more realistic and representative synthetic data. The utility of synthetic data generation extends beyond just addressing class imbalance; it is also highly effective for purposes such as dataset anonymization and missing data imputation. These applications are particularly relevant in scenarios where data privacy is a concern or where incomplete datasets limit the effectiveness of data analysis and model training.

In our work, we have explored a wide range of state-of-the-art techniques in the field of synthetic tabular data generation. Our aim has been to provide guidelines, insights, and a deeper understanding of these advanced techniques. Traditionally, model selection and parameter tuning for synthetic tabular data generation have been non-trivial processes. The development of GenTab addresses this challenge by providing a user-friendly library that offers a wide range of generation models with reasonable default parameters, model selection, tuning, and evaluation. We hope this library encourages and facilitates further efforts towards synthetic tabular generation methods, and their systematic evaluation.

Our findings indicate that DMs, specifically ForestDiffusion and AutoDiffusion, offer the best synthetic tabular data generation alternative if high computational resources are available. ForestDiffusion consistently excels across our tests, achieving top or near-top performance in fidelity, privacy, and ML utility, surpassing the original dataset in most imbalanced classification metrics. In contrast, AEs and specific GAN architectures offer energy-efficient alternatives due to their fast training and generation times. While DMs remain the clear choice for fidelity and ML utility, AEs and some GAN models deliver commendable results, making them in fact feasible alternatives having energy-efficiency in mind. Lastly, when privacy is of paramount importance, GANs emerge as the top performers, the only instance in which DMs have not bested all other methods.

ACKNOWLEDGEMENTS

This work has been supported by Xunta de Galicia: ED431F 2021/11 and ED431G 2023/01. It was also partially supported through the research projects AEI/PID2020-115734RB-C22 and PID2022-136435NB-I00, funded by MCIN/AEI/10.13039/501100011033 and by 'ERDF A way of making Europe', EU. Jose A. Iglesias-Guitian also acknowledges the UDC-Inditex InTalent programme and the Spanish Ministry of Science and Innovation (AEI/RYC2018-025385-I). The authors acknowledge funding for the open access charge provided by Universidade da Coruña/CISUG.

REFERENCES

[1] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis*, 6(5):429–449, 2002.

[2] Nathalie Japkowicz. The class imbalance problem: Significance and strategies. In *Proc. of the International Conference on Artificial Intelligence (IC-AI)*, volume 56, pages 111–117, 2000.

[3] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2008.

[4] Rushi Longadge and Snehalata Dongre. Class imbalance problem in data mining review. *arXiv:1305.1707*, 2013.

[5] Amit Gupta, MC Lohani, and Mahesh Manchanda. Financial fraud detection using naive bayes algorithm in highly imbalance data set. *Journal of Discrete Mathematical Sciences and Cryptography*, 24(5):1559–1572, 2021.

[6] Omar A. Mures, Javier Taibo, Emilio J. Padrón, and Jose A. Iglesias-Guitian. PlayNet: real-time handball play classification with Kalman embeddings and neural networks. *The Visual Computer*, 40(4):2695–2711, 2024.

[7] Shoujin Wang, Wei Liu, Jia Wu, Longbing Cao, Qinxue Meng, and Paul J Kennedy. Training deep neural networks on imbalanced data sets. In *Proc. of the International Joint Conference on Neural Networks (IJCNN)*, pages 4368–4374, 2016.

[8] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):27, 2019.

[9] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proc. of the ICML Workshop on Unsupervised and Transfer Learning*, pages 37–49, 2012.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[11] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proc. of the International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[12] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[13] 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(1):321–357, 2002.

[14] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *Proc. of IEEE International Joint Conference on Neural Networks (IEEE world congress on computational intelligence)*, pages 1322–1328, 2008.

[15] Mitchell Scott and Jo Plested. GAN-SMOTE: A generative adversarial network approach to synthetic minority oversampling. *Australian Journal of Intelligent Information Processing Systems*, 15(2):29–35, 2019.

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

[17] Herkulaas MvE Combrink, Vukosi Marivate, and Benjamin Rosman. Comparing synthetic tabular data generation between a probabilistic model and a deep learning model for education use cases. *arXiv:2210.08528*, 2022.

[18] Ricardo Cardoso Pereira, Miriam Seoane Santos, Pedro Pereira Rodrigues, and Pedro Henriques Abreu. Reviewing autoencoders for missing data imputation: Technical trends, applications and outcomes. *Journal of Artificial Intelligence Research*, 69:1255–1285, 2020.

[19] Alvaro Figueira and Bruno Vaz. Survey on synthetic data generation, evaluation methods and GANs. *Mathematics*, 10(15):2733, 2022.

[20] Joao Fonseca and Fernando Bacao. Tabular and latent space synthetic data generation: a literature review. *Journal of Big Data*, 10(1):115, 2023.

[21] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–21, 2022.

[22] Lasse Hansen, Nabeel Seedat, Mihaela van der Schaar, and Andrija Petrovic. Reimagining synthetic tabular data generation through data-centric AI: A comprehensive benchmark. *Advances in Neural Information Processing Systems*, 36:33781–33823, 2023.

[23] Shriyank Somvanshi, Subasish Das, Syed Aaqib Javed, Gian Antariksa, and Ahmed Hossain. A survey on deep tabular learning. *arXiv:2410.12034*, 2024.

[24] Marko Bohanec. Car Evaluation. UCI Machine Learning Repository, 1997.

[25] Zhuoyuan Zheng, Yunpeng Cai, and Ye Li. Oversampling method for imbalanced classification. *Computing and Informatics*, 34(5):1017–1037, 2015.

[26] Fida K Dankar and Mahmoud Ibrahim. Fake it till you make it: Guidelines for effective synthetic data generation. *Applied Sciences*, 11(5):2158, 2021.

[27] Ping Li, Tong Li, Heng Ye, Jin Li, Xiaofeng Chen, and Yang Xiang. Privacy-preserving machine learning with multiple data providers. *Future Generation Computer Systems*, 87:341–350, 2018.

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

[29] Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. CTAB-GAN: Effective table data synthesizing. In *Proc. of the Asian Conference on Machine Learning*, pages 97–112. PMLR, 2021.

[30] Zilong Zhao, Aditya Kunar, Robert Birke, Hiek Van der Scheer, and Lydia Y Chen. CTAB-GAN+: Enhancing tabular data synthesis. *Frontiers in Big Data*, 6:1296508, 2024.

[31] Alexia Jolicoeur-Martineau, Kilian Fatras, and Tal Kachman. Generating and imputing tabular data via diffusion and flow-based gradient-boosted trees. In *Proc. of the International Conference on Artificial Intelligence and Statistics*, pages 1288–1296. PMLR, 2024.

[32] Namjoon Suh, Xiaofeng Lin, Din-Yin Hsieh, Merhdad Honarkhah, and Guang Cheng. AutoDiff: combining auto-encoder and diffusion model for tabular data synthesizing. *arXiv:2310.15479*, 2023.

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

[34] Zilong Zhao, Robert Birke, and Lydia Chen. TabuLa: Harnessing language models for tabular data synthesis. *arXiv:2310.12746*, 2023.

[35] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[36] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013.

[37] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[38] Zhiqiang Wan, Yazhou Zhang, and Haibo He. Variational autoencoder based synthetic data generation for imbalanced learning. In *2017 IEEE symposium series on computational intelligence (SSCI)*, pages 1–7. IEEE, 2017.

[39] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook*, pages 353–374, 2023.

[40] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27, 2014.

[41] Zhengwei Wang, Qi She, and Tomas E Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[42] Vignesh Sampath, Iñaki Maurtua, Juan Jose Aguilar Martin, and Aitor Gutierrez. A survey on generative adversarial networks for imbalance problems in computer vision tasks. *Journal of Big Data*, 8:1–59, 2021.

[43] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv:1411.1784*, 2014.

[44] Giovanni Mariani, Florian Scheidegger, Roxana Istrate, Costas Bekas, and Cristiano Malossi. BAGAN: Data augmentation with balancing GAN. *arXiv:1803.09655*, 2018.

[45] Justin Engelmann and Stefan Lessmann. Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning. *Expert Systems with Applications*, 174:114582, 2021.

[46] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *Proc. of the International Conference on Machine Learning*, pages 2642–2651. PMLR, 2017.

[47] Qian Zhou and Bo Sun. A gaussian-based wgan-gp oversampling approach for solving the class imbalance problem. *International Journal of Applied Mathematics and Computer Science*, 34(2), 2024.

[48] Peter Xue-Kun Song. Multivariate dispersion models generated from Gaussian copula. *Scandinavian Journal of Statistics*, 27(2):305–320, 2000.

[49] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proc. of the International Conference on Machine Learning*, pages 214–223. PMLR, 2017.

[50] Rick Sauber-Cole and Taghi M Khoshgoftaar. The use of generative adversarial networks to alleviate class imbalance in tabular data: a survey. *Journal of Big Data*, 9(1): 98, 2022.

[51] Yiming Qin, Huangjie Zheng, Jiangchao Yao, Mingyuan Zhou, and Ya Zhang. Class-balancing diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18434–18443, 2023.

[52] Chenghao Xu, Jiexi Yan, Muli Yang, and Cheng Deng. Rethinking noise sampling in class-imbalanced diffusion models. *IEEE Transactions on Image Processing*, 2024.

[53] Ziyi Chang, George A Koulieris, and Hubert PH Shum. On the design fundamentals of diffusion models: A survey. *arXiv:2306.04542*, 2023.

[54] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[55] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv:2011.13456*, 2020.

[56] 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.

[57] Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. TabDDPM: Modelling tabular data with diffusion models. In *Proc. of the International Conference on Machine Learning*, pages 17564–17579. PMLR, 2023.

[58] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005.

[59] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 115, pages 574–584. PMLR, 2020.

[60] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34:1415–1428, 2021.

[61] Fan Pu Zeng and Owen Wang. Score-based diffusion models. *fanpu.io*, Jun 2023.

[62] Jayoung Kim, Chaejeong Lee, Yehjin Shin, Sewon Park, Minjung Kim, Noseong Park, and Jihoon Cho. SOS: Score-based oversampling for tabular data. In *Proc. of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 762–772, 2022.

[63] Jayoung Kim, Chaejeong Lee, and Noseong Park. STaSy: Score-based tabular data synthesis. In *Proc. of the International Conference on Learning Representations*, 2023.

[64] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, 2022.

[65] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[66] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4):1–39, 2023.

[67] Heejoon Koo and To Eun Kim. A comprehensive survey on generative diffusion models for structured data. *ArXiv, abs/2306.04139 v2*, 2023.

[68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[69] Tomáš Mikolov, Anoop Deoras, Daniel Povey, Lukáš Burget, and Jan Černockỳ. Strategies for training large scale neural network language models. In *Proc. of the IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 196–201, 2011.

[70] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[71] Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander J Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. Large language model as attributed training data generator: A tale of diversity and bias. *Advances in Neural Information Processing Systems*, 36, 2024.

[72] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv:2001.08361*, 2020.

[73] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv:2206.07682*, 2022.

[74] Nabeel Seedat, Nicolas Huynh, Boris van Breugel, and Mihaela van der Schaar. Curated LLM: Synergy of LLMs and data curation for tabular augmentation in ultra low-data regimes. *arXiv:2312.12112*, 2023.

[75] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv:2303.08774*, 2023.

[76] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv:2303.18223*, 2023.

[77] Xi Fang, Weijie Xu, Fiona Anting Tan, Jiani Zhang, Ziqing Hu, Yanjun (Jane) Qi, Scott Nickleach, Diego Socolinsky, "SHS" Srinivasan Sengamedu, and Christos Faloutsos. Large language models (LLMs) on tabular data: Prediction, generation, and understanding - A survey. *Transactions on Machine Learning Research*, 2024.

[78] Felipe Pezoa, Juan L Reutter, Fernando Suarez, Martín Ugarte, and Domagoj Vrgoč. Foundations of JSON schema. In *Proc. of the International Conference on World Wide Web*, pages 263–273, 2016.

[79] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. 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.

[80] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[81] Jianhua Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37 (1):145–151, 1991.

[82] Aaditya Ramdas, Nicolás García Trillos, and Marco Cuturi. On Wasserstein two-sample testing and related families of nonparametric tests. *Entropy*, 19(2):47, 2017.

[83] Philip Sedgwick. Pearson's correlation coefficient. *BMJ*, 345, 2012.

[84] Raymond M Leuthold. On the use of Theil's inequality coefficients. *American Journal of Agricultural Economics*, 57(2):344–346, 1975.

[85] Samson Otieno Ooko, Didacienne Mukanyiligira, Jean Pierre Munyampundu, and Jimmy Nsenga. Synthetic exhaled breath data-based edge AI model for the prediction of chronic obstructive pulmonary disease. In *Proc. of the International Conference on Computing and Communications Applications and Technologies (I3CAT)*, pages 1–6, 2021.

[86] Chao Yan, Ziqi Zhang, Steve Nyemba, and Bradley A Malin. Generating electronic health records with multiple data types and constraints. In *Proc. of the AMIA annual symposium*, volume 2020, page 1335, 2020.

[87] Jinsung Yoon, Lydia N Drumright, and Mihaela Van Der Schaar. Anonymization through data synthesis using generative adversarial networks (ADS-GAN). *IEEE Journal of Biomedical and Health Informatics*, 24(8): 2378–2388, 2020.

[88] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 30, 2017.

[89] 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.

[90] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27, 2011.

[91] 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.

[92] Brian W Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.

[93] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery & Data mining*, pages 2623–2631, 2019.

[94] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in Neural Information Processing Systems*, 24, 2011.

[95] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[96] Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Proc. of the Artificial Intelligence and Statistics*, pages 240–248. PMLR, 2016.

[97] Omar A Mures, Javier Taibo, Emilio J Padrón, and Jose A Iglesias-Guitian. A comprehensive handball dynamics dataset for game situation classification. *Data in Brief*, 52:109848, 2024.

[98] Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996.

[99] Zejin Ding. Diversified ensemble classifiers for highly imbalanced data learning and their application in bioinformatics, 2011.

[100] Ross Quinlan. Thyroid disease. UCI Machine Learning Repository, 1986.

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

[102] Mushroom. UCI Machine Learning Repository, 1981.

[103] Kenta Nakai. Ecoli. UCI Machine Learning Repository, 1996.

[104] Stanisław Węglarczyk. Kernel density estimation and its application. In *ITM web of conferences*, volume 23, 2018.

[105] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional GANs. *arXiv:1706.02633*, 2017.

[106] 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.

Table 8: MCC, accuracy, precision, recall, and F-Score for the tested methods in the **Car Evaluation** dataset.

Model	MCC	Acc.	Precision		Recall		F-Score	
			Weighted	Macro	Weighted	Macro	Weighted	Macro
None	0.74	98.8%	98.1%	85.5%	98.8%	88.1%	98.4%	86.8%
SMOTE [13]	0.88	98.9%	99.2%	90.3%	98.9%	**98.6**%	99.0%	93.7%
ADASYN [14]	0.89	98.9%	99.2%	90.7%	98.9%	**98.6**%	99.0%	93.9%
TVAE [28]	**0.90**	**99.1**%	**99.3**%	**93.2**%	**99.1**%	97.4%	**99.2**%	**94.8**%
CTGAN [28]	0.70	92.4%	98.2%	80.5%	92.4%	91.9%	94.4%	81.9%
GaussianCopula [16]	0.70	93.9%	98.2%	78.0%	93.9%	95.6%	95.4%	82.2%
CopulaGAN [28]	0.76	93.3%	98.6%	83.9%	93.3%	94.4%	95.2%	85.2%
CTAB-GAN [29]	0.73	95.5%	98.3%	81.8%	95.5%	93.5%	96.5%	84.5%
CTAB-GAN+ [30]	0.71	95.0%	98.2%	80.2%	95.0%	93.6%	96.1%	83.2%
AutoDiffusion [32]	0.89	99.0%	**99.3**%	91.6%	99.0%	98.2%	99.1%	94.2%
ForestDiffusion [31]	0.87	98.7%	99.1%	89.6%	98.7%	98.1%	98.9%	93.0%
GReaT [33]	0.20	52.5%	96.5%	54.0%	52.5%	75.3%	64.5%	40.6%
Tabula [34]	0.37	82.3%	96.7%	63.1%	82.3%	84.5%	87.4%	60.1%

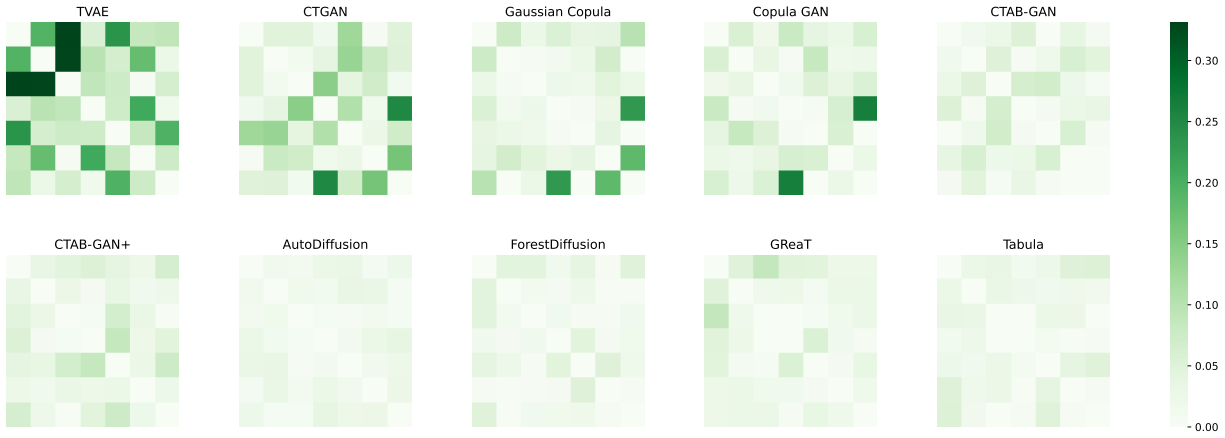

Figure 11: Heatmaps of the pair-wise correlation of the synthetic versus the original data in the **Car Evaluation** dataset. Pixels represent the divergence between the synthetic and real feature correlations. Ideally, the synthetic data should have the same correlation between features as the real data. This would be equivalent to a heatmap with all white. Lighter colors indicate that the synthetic data is better at replicating the real data (lighter is better).

Appendix

A CAR EVALUATION

We have evaluated the imbalanced version of the UCI Car Evaluation dataset [99], composed exclusively of categorical features. This dataset exhibits one of the highest imbalance ratios among our selected datasets, providing a good testbed for synthetic data generation methods.

ML Utility In the analysis presented in Table 8, TVAE stands out by achieving an average MCC of 0.9, which surpasses the real dataset by a wide margin, 0.16. Notably, AutoDiffusion, SMOTE and ADASYN also show approximately the same result. However, a deeper dive into metrics such as, accuracy, precision, recall, and F-Score reveals that TVAE clearly outperforms other selected methods. TVAE obtained the best results in the majority of metrics, except for macro recall, trailing SMOTE and ADASYN by a small margin. Another relevant improvement was achieved in the macro F-Score metric, specially important for imbalanced datasets, where it has obtained a ~94.8 %, yielding an advantage over the original dataset of ~8 %. AutoDiffusion closely followed TVAE, either tying or slightly trailing in most metrics, again proving its effectiveness. A notable observation from this analysis is the inability of GAN and LLM methods to surpass the original dataset in terms of MCC and other relevant metrics.

Fidelity We opted for utilizing the original UCI Car Evaluation dataset [24], which is strictly categorical and not binarized, allowing us to more accurately assess the preservation of global structure and better ascertain correlations among features. The findings, presented in Fig. 11, highlight that DMs, particularly AutoDiffusion, tend to outshine other methods when trying to preserve global structure in categorical datasets. Interestingly, LLMs also show commendable performance in this context, in contrast with the results obtained when using binarized data. Yet, it is important to note that this superior performance in structure preservation does not always translate into enhanced ML performance, as we can see in previous results. TVAE, when not using the binarized dataset, does quite poorly in terms of preserving pair-wise correlations, together with CTGAN and CopulaGAN.

Table 9: MCC, accuracy, precision, recall, and F-Score for the tested methods in the **PlayNet** dataset.

Model	MCC	Acc.	Precision		Recall		F-Score	
			Weighted	Macro	Weighted	Macro	Weighted	Macro
None	**0.73**	**88.8%**	**89.3%**	**78.4%**	**88.8%**	82.3%	**88.9%**	**79.7%**
SMOTE [13]	0.68	82.2%	88.6%	71.6%	82.2%	82.6%	84.4%	73.9%
ADASYN [14]	0.68	80.7%	88.9%	72.3%	80.7%	82.4%	83.4%	72.9%
TVAE [28]	0.70	84.2%	88.7%	75.4%	84.2%	81.4%	85.9%	76.8%
CTGAN [28]	0.72	88.5%	89.1%	76.6%	88.5%	81.9%	88.7%	78.3%
GaussianCopula [16]	0.72	88.4%	89.1%	76.7%	88.4%	81.6%	88.7%	78.4%
CopulaGAN [28]	0.71	86.4%	88.9%	74.9%	86.4%	82.2%	87.4%	77.5%
CTAB-GAN [29]	0.69	84.6%	88.8%	71.0%	84.6%	82.0%	86.2%	74.0%
CTAB-GAN+ [30]	0.64	80.8%	88.0%	65.7%	80.8%	81.0%	83.5%	68.3%
AutoDiffusion [32]	0.72	88.2%	89.1%	77.3%	88.2%	82.2%	88.5%	78.9%
ForestDiffusion [31]	0.70	83.5%	89.0%	76.0%	83.5%	82.9%	85.5%	77.5%
GReaT [33]	0.68	82.4%	88.9%	73.9%	82.4%	**83.7%**	84.7%	75.2%
Tabula [34]	0.67	82.5%	88.7%	71.7%	82.5%	82.6%	84.8%	72.8%

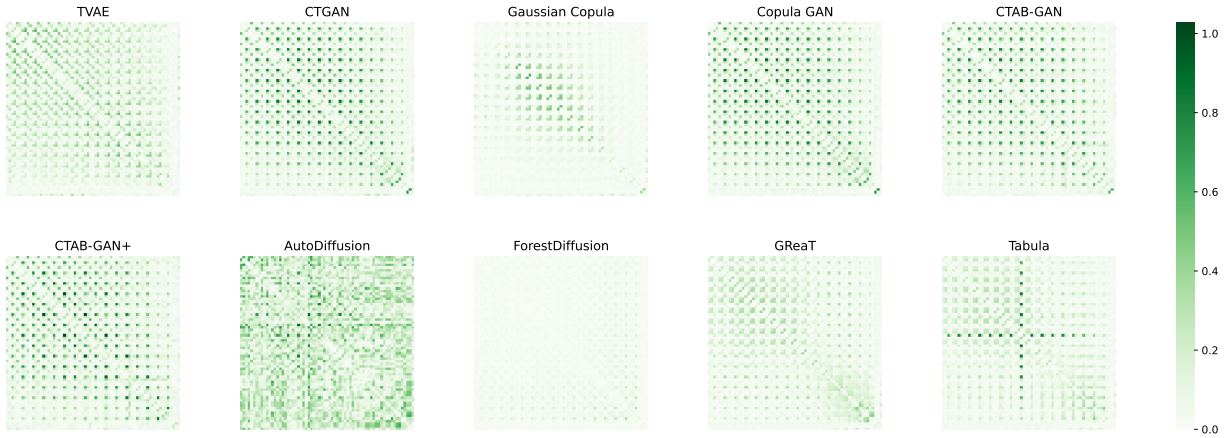

Figure 12: Heatmaps of the pair-wise correlation of the synthetic data versus the original data in the **PlayNet** dataset. Pixels represent the divergence between the synthetic and real feature correlations. Ideally, the synthetic data should have the same correlation between features as the real data. This would be equivalent to a heatmap with all white. Lighter colors indicate that the synthetic data is better at replicating the real data (lighter is better).

On the other hand, CTAB-GAN found a middle ground, performing neither at the top nor at the bottom, while CTAB-GAN+ again faced difficulties when generating samples for minority classes, failing to complete the task. In summary, DMs have reasserted their dominance in preserving global-structure, showing their capacity to maintain complex variable relationships, rivaling LLMs in probably the most favorable dataset for them. While their proficiency in structure preservation is evident, as with the other datasets, it is important to recognize that this alone does not guarantee superior ML utility.

B PLAYNET

The PlayNet dataset serves as a unique scenario for evaluating generative models, given its composition, which includes player positions and velocities in the game court. Due to the nature of this dataset, we provide additional data compared to others.

ML Utility Our findings, as detailed in Table 9, indicate that in the context of the PlayNet dataset, synthetic data did not outperform the real dataset across most ML utility metrics. The methods that came close to the performance of the original dataset regarding MCC were AutoDiffusion, CTGAN, and GaussianCopula, all trailing the baseline MCC by a narrow margin of just 0.01. Notably, GReaT managed to outperform the real dataset in terms of macro recall, with a ~83.7 %, surpassing by a ~1.4 % the baseline number. LLM methods struggled, apart from the aforementioned exception. The inherent structure of LLMs, optimized for tokenizing and processing textual information, encounters challenges with numerical data due to the tokenization overhead and the repetitive nature of the data. These properties pose a big overhead when tokenizing rows and performing inference, due to each column consuming a high amount of tokens, as patterns are seldom repeated. These results coupled with the consistent increase in recall performance for several models, show that even in unfavourable scenarios, synthetic datasets can be beneficial for decreasing the number of cases in which downstream models miss scarce situations. Conversely, while demonstrating acceptable performance, SMOTE and ADASYN exhibited reduced effectiveness compared to network-based approaches.

Fidelity In Fig. 12, the cross-correlation chart reveals the effectiveness of ForestDiffusion when preserving global feature relationships in this dataset. Conversely, AutoDiffusion exhibits the weakest performance, with several variables failing to maintain correlations present in the original dataset. Interestingly, Tabula, while demonstrating a high level of preserved correlation, underperforms AutoDiffusion in almost every ML utility metric. These results suggest that a high level of correlation does not necessarily guarantee improved ML performance, as we saw in the previous dataset.

Furthermore, Fig. 13 depicts randomly chosen real game situations alongside the synthetic samples generated by each method. Since the dataset samples equate to real positions and velocities in a handball arena, this visualization allows us to analyze player positions and velocities from a qualitative perspective, complementing our quantitative evaluations. Consistent with the prior heatmap analysis, ForestDiffusion exhibits the strongest performance, accurately depicting player positions and velocities across most game situations. Conversely, AutoDiffusion displays the most significant deviations from realistic player data. GReaT merits particular attention, as it visually replicates player behavior well in most situations, with the exception of a player exhibiting unrealistic speed during a penalty. Several methods, including ForestDiffusion, GReaT, and Tabula, generate visually plausible player positions and velocities during timeouts, accurately reflecting the lack of player movement in such situations. Transitions reveal a broader range of methods achieving visually plausible outcomes, including CTAB-GAN, ForestDiffusion, and GReaT. Accurately depicting attack scenarios remains one of the most challenging tasks, yet ForestDiffusion and GReaT appear to generate the most realistic player dynamics. Penalties pose significant challenges for all methods, with ForestDiffusion emerging as the most plausible, although GReaT presents a potentially viable alternative if not for a few unnatural player positions and velocities as mentioned above.

Figure 13: Comparison of court snapshots of the real and synthetic handball situations for each model trained with the **PlayNet** dataset. Player positions are represented by blue dots and velocities by orange arrows, representing player dynamics. ForestDiffusion demonstrates the most accurate replication of real game situations, while AutoDiffusion, CTAB-GAN, and CTAB-GAN+ exhibit the weakest performance, with unrealistic player speeds and positions.

C ADULT

The UCI Adult dataset, characterized by its diverse features, provides an ideal opportunity to assess the performance of LLM methods. While these methods performed well, they did not demonstrate a clear outperformance over other models. This outcome suggests that while LLMs are promising in handling varied data types, their effectiveness can be context-dependent and may not always translate to superior performance in every scenario.

ML Utility The analysis of the UCI Adult dataset (see Table 10) reveals that several synthetic datasets achieved comparable or superior performance to the real dataset in terms of MCC. The CTAB-GAN family of models emerged as top performers, achieving an MCC of 0.59, an improvement of 0.02 over the real dataset. Additionally they boast the highest weighted precision ~85.6 %, surpassing the original data by ~0.4 %. GaussianCopula also performed well in this dataset, particularly in F-Score, exceeding the real dataset in macro and weighted scores. It achieved a weighted F-Score of ~85 % and a macro F-Score of ~78.7 %, representing a ~0.7 % and a ~1.6 % improvement respectively. DMs and LLMs demonstrated competitive performance, matching the original dataset in MCC and even surpassing it in certain metrics, such as macro recall and macro F-score.

Fidelity Regarding the preservation of global relationships within the data, CTAB-GAN variants, and AutoDiffusion have emerged as the most effective in our tests, this is depicted in the provided cross-correlation chart (see Fig. 14). Tabula presents a compelling example, illustrating that strong preservation of global relationships does not necessarily improve ML utility. Despite showing good results in global relationship preservation, its ML utility did not surpass that of GReaT, which, although obtaining slightly weaker fidelity results, outperformed it. Conversely, methods such as TVAE, CTGAN, or CopulaGAN displayed weaker performance, as seen in their respective heatmaps. In these instances, it translated into less favorable outcomes in ML utility tests. These results suggest that fidelity is sometimes not indicative of ML utility, and that results may vary across different generative models and use cases, highlighting the importance of proper model selection and tuning for maximizing ML utility.

D ECOLI

The Ecoli dataset, with its limited sample size, provides a valuable testbed for evaluating the efficacy of synthetic data generation methods in data-scarce scenarios. As anticipated, this dataset highlights the strengths of network-based oversampling techniques, which demonstrate superior performance in augmenting limited data and improving downstream model accuracy. Moreover, its biological context offers a unique opportunity to assess the applicability of these methods in a domain where data scarcity is often a significant challenge.

ML Utility Table 11 presents the results for this dataset. DMs, particularly ForestDiffusion, again proved to have exceptional performance when enhancing imbalanced datasets. ForestDiffusion surpasses all other methods across almost every metric, achieving a remarkable MCC score of 0.75 (a 0.31 improvement over the original dataset) and a macro F-score of approximately ~87.2 % (a ~16.3 % improvement). Notably, ForestDiffusion even outperforms the original dataset in some non-imbalanced metrics. TVAE, representing AE-based methods, secures the second position but trails ForestDiffusion considerably. Nevertheless, it still improves upon the original dataset's MCC by 0.21 and macro F-score by approximately ~9.8 %. GANs and LLMs also surpass the original dataset in imbalanced metrics, along with local methods like SMOTE and ADASYN, reinforcing the observation that synthetic data generation is particularly effective for smaller datasets.

Fidelity Fig. 15 presents cross-correlation heatmaps, illustrating the ability of surveyed models to capture underlying relationships within the data. Tabula demonstrates the closest resemblance to the original dataset's correlation structure, in this instance translating into improved ML performance. AutoDiffusion closely follows, also effectively capturing underlying variable relationships and obtaining low correlation distances. ForestDiffusion, consistent with its strong performance across datasets, maintains good results with low divergence in pairwise correlations, although not the best in this case. AE based methods also perform well, although some loss of correlation

Table 10: MCC, accuracy, precision, recall, and F-Score for the tested methods in the **Adult** dataset.

Model	MCC	Acc.	Precision		Recall		F-Score	
			Weighted	Macro	Weighted	Macro	Weighted	Macro
None	0.57	**85.6**%	85.2%	**82.9**%	**85.6**%	75.2%	84.3%	77.1%
SMOTE [13]	0.57	83.2%	84.7%	78.8%	83.2%	78.4%	83.5%	77.9%
ADASYN [14]	0.57	81.2%	85.0%	76.2%	81.2%	**81.1**%	82.2%	77.3%
TVAE [28]	0.57	83.6%	84.7%	78.7%	83.6%	78.9%	83.8%	78.3%
CTGAN [28]	0.58	82.7%	85.0%	77.8%	82.7%	80.1%	83.3%	78.1%
GaussianCopula [16]	0.58	85.5%	85.0%	81.5%	85.5%	77.0%	**85.0**%	**78.7**%
CopulaGAN [28]	0.57	81.9%	84.6%	76.4%	81.9%	80.4%	82.7%	77.5%
CTAB-GAN [29]	**0.59**	83.9%	85.4%	79.8%	83.9%	79.2%	84.1%	78.6%
CTAB-GAN+ [30]	**0.59**	83.2%	**85.6**%	79.7%	83.2%	79.4%	83.5%	78.2%
AutoDiffusion [32]	0.57	82.3%	84.7%	76.8%	82.3%	80.3%	83.0%	77.8%
ForestDiffusion [31]	0.57	84.7%	84.6%	80.4%	84.7%	76.8%	84.3%	78.1%
GReaT [33]	0.57	83.2%	84.4%	77.5%	83.2%	79.3%	83.6%	78.1%
Tabula [34]	0.57	82.3%	84.6%	77.1%	82.3%	79.7%	82.9%	77.6%

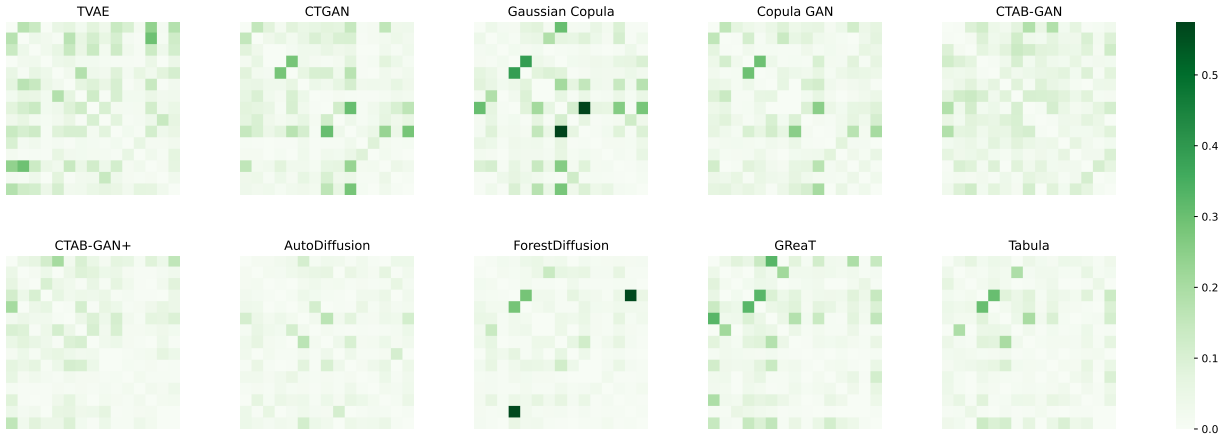

Figure 14: Heatmaps of the pair-wise correlation of the synthetic data versus the original data in the **Adult** dataset. Pixels represent the divergence between the synthetic and real feature correlations. Ideally, the synthetic data should have the same correlation between features as the real data. This would be equivalent to a heatmap with all white. Lighter colors indicate that the synthetic data is better at replicating the real data (lighter is better).

Table 11: MCC, accuracy, precision, recall, and F-Score for the tested methods in the **Ecoli** dataset.

Model	MCC	Acc.	Precision		Recall		F-Score	
			Weighted	Macro	Weighted	Macro	Weighted	Macro
None	0.44	93.3%	89.5%	72.0%	93.3%	70.4%	91.3%	70.9%
SMOTE [13]	0.61	90.1%	93.5%	76.3%	90.1%	86.3%	91.1%	79.0%
ADASYN [14]	0.62	90.3%	93.7%	77.0%	90.3%	86.4%	91.3%	79.5%
TVAE [28]	0.65	90.6%	94.2%	77.4%	90.6%	**88.8%**	91.7%	80.7%
CTGAN [28]	0.62	89.6%	93.7%	75.9%	89.6%	87.5%	90.8%	78.9%
GaussianCopula [16]	0.59	94.5%	92.4%	80.6%	94.5%	77.7%	93.3%	78.8%
CopulaGAN [28]	0.60	91.1%	93.3%	79.0%	91.1%	82.5%	91.5%	78.2%
CTAB-GAN [29]	0.47	85.7%	91.1%	68.4%	85.7%	80.2%	87.4%	71.0%
CTAB-GAN+ [30]	0.39	83.2%	89.7%	64.3%	83.2%	76.6%	85.4%	66.8%
AutoDiffusion [32]	0.60	90.4%	93.1%	75.8%	90.4%	85.1%	91.3%	78.6%
ForestDiffusion [31]	**0.75**	**95.1%**	**95.4%**	**87.0%**	**95.1%**	87.7%	**95.2%**	**87.2%**
GReaT [33]	0.38	73.1%	90.7%	62.2%	73.1%	79.1%	78.3%	61.4%
Tabula [34]	0.54	86.2%	92.6%	71.2%	86.2%	84.9%	88.2%	74.0%

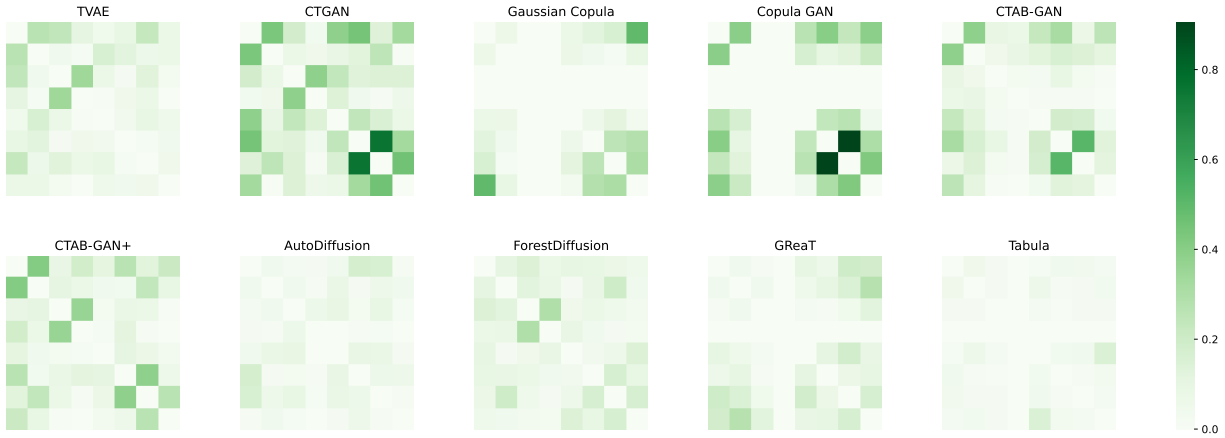

Figure 15: Heatmaps of the pair-wise correlation of the synthetic versus the original data in the **Ecoli** dataset. Pixels represent the divergence between the synthetic and real feature correlations. Ideally, the synthetic data should have the same correlation between features as the real data. This would be equivalent to a heatmap with all white. Lighter colors indicate that the synthetic data is better at replicating the real data (lighter is better).

Table 12: MCC, accuracy, precision, recall, and F-Score for the tested methods in the **Sick Euthyroid** dataset.

Model	MCC	Acc.	Precision		Recall		F-Score	
			Weighted	Macro	Weighted	Macro	Weighted	Macro
None	0.65	**95.5**%	94.0%	80.2%	**95.5**%	84.0%	94.7%	82.0%
SMOTE [13]	0.67	88.4%	95.3%	79.0%	88.4%	**89.9**%	90.3%	80.6%
ADASYN [14]	0.67	86.8%	**95.5**%	79.4%	86.8%	89.5%	88.9%	79.9%
TVAE [28]	0.64	89.9%	94.5%	78.0%	89.9%	87.4%	91.4%	80.3%
CTGAN [28]	0.64	85.4%	94.8%	77.6%	85.4%	86.9%	87.4%	77.7%
GaussianCopula [16]	0.63	84.4%	94.9%	75.6%	84.4%	88.6%	86.6%	76.4%
CopulaGAN [28]	0.67	90.4%	94.8%	80.6%	90.4%	87.3%	91.8%	82.0%
CTAB-GAN [29]	0.57	80.9%	94.2%	73.6%	80.9%	84.6%	82.8%	72.4%
CTAB-GAN+ [30]	0.59	82.0%	94.5%	73.8%	82.0%	87.0%	84.1%	73.8%
AutoDiffusion [32]	**0.70**	95.1%	95.0%	**84.4**%	95.1%	85.6%	**95.0**%	**84.9**%
ForestDiffusion [31]	0.65	86.9%	95.2%	77.6%	86.9%	89.5%	89.1%	78.9%
GReaT [33]	0.43	77.2%	92.0%	64.6%	77.2%	81.7%	81.3%	65.0%
Tabula [34]	-	-	-	-	-	-	-	-

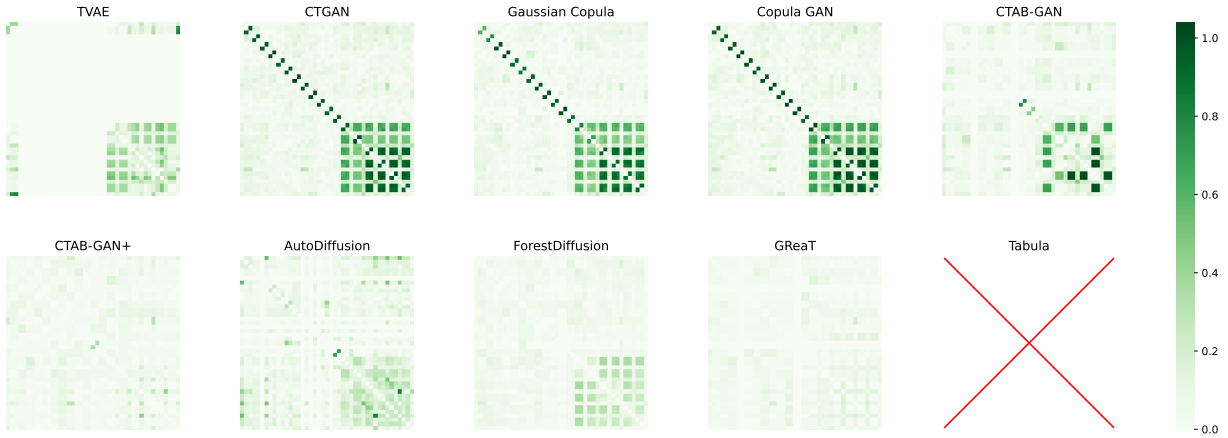

Figure 16: Heatmaps of the pair-wise correlation of the synthetic versus the original data in the **Sick Euthyroid** dataset. Pixels represent the divergence between the synthetic and real feature correlations. Ideally, the synthetic data should have the same correlation between features as the real data. This would be equivalent to a heatmap with all white. Lighter colors indicate that the synthetic data is better at replicating the real data (lighter is better). Methods denoted by a red cross (✗) were unable to generate a sufficient number of samples for the target classes.

is evident. In contrast, GAN-based methods struggle with this dataset, yielding suboptimal results in capturing and preserving data dependencies.

E SICK EUTHYROID

As our first medical dataset, Sick Euthyroid provides a representative example of disease diagnosis scenarios. Its small size and relatively high imbalance ratio are characteristic of medical datasets, particularly those related to thyroid gland disorders. This dataset allows us to evaluate how effectively generative models address challenges specific to the medical domain, where accurate classification of rare conditions is crucial.

ML Utility Table 12 presents the results for the Sick Euthyroid dataset. AutoDiffusion emerges as the clear winner, outperforming all other models in MCC and F-score, further solidifying the strong performance of Diffusion Models. It achieves a 0.05 lead in MCC and a ~2.9 % lead in macro F-score. CopulaGAN, SMOTE, and ADASYN follow with the second-best results, with SMOTE and ADASYN excelling in macro recall, achieving a lead of up to ~5.9 %. LLMs, however, struggle with this dataset, with Tabula failing to generate samples and GReaT underperforming the original data.

Fidelity Fig. 16 depicts the pairwise correlation distances between the original and synthetic dataset, providing insights into the ability of different generative models for preserving variable relationships. GReaT achieves the best performance in this assessment, closely replicating the correlation structure of the original data. However, this fidelity does not directly translate to superior performance in oversampling for ML utility, where GReaT did not perform well. CTAB-GAN+ also demonstrates strong performance in preserving variable relationships. In contrast, some other GAN-based methods (CTGAN and CopulaGAN) and GaussianCopula exhibit weaker performance, notably deviating from the original dataset's correlation structure.

Table 13: MCC, accuracy, precision, recall, and F-Score for the tested methods in the **California Housing** dataset.

Model	MCC	Acc.	Precision		Recall		F-Score	
			Weighted	Macro	Weighted	Macro	Weighted	Macro
None	0.46	57.5%	54.9%	51.9%	57.5%	48.8%	55.1%	49.0%
SMOTE [13]	0.45	54.6%	55.8%	50.2%	54.6%	**53.6%**	54.0%	50.6%
ADASYN [14]	0.45	55.8%	56.1%	51.8%	55.8%	50.1%	54.1%	47.6%
TVAE [28]	0.45	57.0%	53.4%	49.1%	57.0%	47.2%	54.1%	46.9%
CTGAN [28]	0.45	57.1%	53.5%	49.7%	57.1%	47.0%	54.0%	46.9%
GaussianCopula [16]	0.44	56.6%	53.5%	49.6%	56.6%	46.5%	53.8%	46.6%
CopulaGAN [28]	0.45	56.8%	53.0%	48.6%	56.8%	46.4%	53.6%	46.0%
CTAB-GAN [29]	**0.47**	57.4%	**57.4%**	**54.6%**	57.4%	51.3%	56.4%	**51.5%**
CTAB-GAN+ [30]	**0.47**	**57.7%**	57.2%	53.7%	**57.7%**	51.7%	**56.5%**	51.4%
AutoDiffusion [32]	0.44	54.2%	55.6%	48.9%	54.2%	51.9%	53.9%	49.2%
ForestDiffusion [31]	0.46	56.5%	56.0%	52.2%	56.5%	50.2%	54.8%	49.6%
GReaT [33]	0.40	47.9%	48.4%	43.7%	47.9%	46.0%	47.9%	44.5%
Tabula [34]	0.41	48.1%	48.6%	44.0%	48.1%	45.8%	48.1%	44.5%

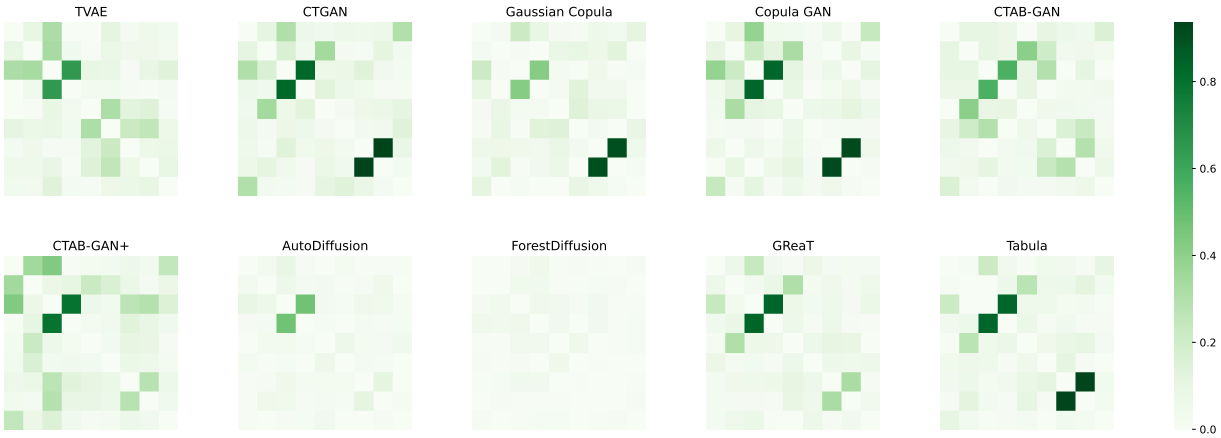

Figure 17: Heatmaps of the pair-wise correlation of the synthetic versus the original data in the **California Housing** dataset. Pixels represent the divergence between the synthetic and real feature correlations. Ideally, the synthetic data should have the same correlation between features as the real data. This would be equivalent to a heatmap with all white. Lighter colors indicate that the synthetic data is better at replicating the real data (lighter is better).

F CALIFORNIA HOUSING

The California Housing dataset, derived from the 1990 US Census, offers a unique and valuable opportunity to assess the capabilities of synthetic data generation methods beyond typical tabular datasets. It provides median house values for California districts in hundreds of thousands of dollars ($100 000), but it also includes geographic information in the form of latitude and longitude coordinates. This spatial component allows for a deeper evaluation of generative models, testing their ability to not only reproduce individual feature distributions but also to capture the complex spatial relationships and dependencies inherent in real-world geographical data.

ML Utility Table 13 presents the results for the California Housing dataset. The CTAB-GAN family of models obtained the best performance, achieving a 0.01 increase in MCC, a ~2.7 % increase in macro precision, and a ~2.5 % increase in macro F-score over the original dataset. SMOTE leads in macro recall, surpassing the original dataset by ~4.8 %. Despite strong fidelity performance, DMs show less impressive results in oversampling, only marginally exceeding the original dataset in macro recall, precision, and F-score. AEs, represented by TVAE, and LLMs underperformed across all metrics.

Fidelity Fig. 17 illustrates the pairwise correlation distances between the original California Housing dataset and the synthetic datasets generated by the chosen models. ForestDiffusion demonstrates exceptional performance, accurately capturing the correlation structure of the original data and reinforcing its strong results observed in the main paper's scatterplot of price versus latitude and longitude. If we account for the ML utility results, we can see that the high fidelity does not translate into good data augmentation performance, probably due to not having enough differences between datasets for capturing new unseen data patterns. AutoDiffusion also performs well, with only minor discrepancies in two pairs of variables. These results reinforce that in terms of fidelity, DMs are the best option. In contrast, other models struggle to effectively preserve variable relationships. This is particularly evident in LLMs, which, despite capturing some geographical relationships well, exhibit significant deviations in other pair-wise correlations. GAN models struggle the most to maintain accurate correlations. AEs,

Table 14: MCC, accuracy, precision, recall, and F-Score for the tested methods in the **Mushroom** dataset.

Model	MCC	Acc.	Precision		Recall		F-Score	
			Weighted	Macro	Weighted	Macro	Weighted	Macro
None	0.96	98.0%	98.1%	98.1%	98.0%	97.9%	97.9%	97.9%
SMOTE [13]	**0.98**	99.0%	99.0%	99.0%	99.0%	99.0%	99.0%	99.0%
ADASYN [14]	0.97	98.2%	98.3%	98.3%	98.2%	98.2%	98.2%	98.2%
TVAE [28]	**0.98**	**99.1%**	**99.1%**	**99.1%**	**99.1%**	**99.1%**	**99.1%**	**99.1%**
CTGAN [28]	0.97	98.6%	98.6%	98.6%	98.6%	98.6%	98.6%	98.6%
GaussianCopula [16]	0.97	98.6%	98.6%	98.7%	98.6%	98.6%	98.6%	98.6%
CopulaGAN [28]	0.97	98.7%	98.7%	98.7%	98.7%	98.7%	98.7%	98.7%
CTAB-GAN [29]	**0.98**	99.0%	99.0%	99.0%	99.0%	99.0%	99.0%	99.0%
CTAB-GAN+ [30]	**0.98**	99.0%	99.0%	99.0%	99.0%	99.0%	99.0%	99.0%
AutoDiffusion [32]	0.97	98.7%	98.7%	98.7%	98.7%	98.7%	98.7%	98.7%
ForestDiffusion [31]	0.97	98.6%	98.6%	98.6%	98.6%	98.6%	98.6%	98.6%
GReaT [33]	0.80	79.8%	79.8%	79.8%	79.8%	79.8%	79.8%	79.8%
Tabula [34]	0.80	80.0%	80.0%	80.0%	80.0%	80.0%	80.0%	80.0%

Figure 18: Heatmaps of the pair-wise correlation of the synthetic versus the original data in the **Mushroom** dataset. Pixels represent the divergence between the synthetic and real feature correlations. Ideally, the synthetic data should have the same correlation between features as the real data. This would be equivalent to a heatmap with all white. Lighter colors indicate that the synthetic data is better at replicating the real data (lighter is better).

and by extension TVAE, obtain reasonable performance when preserving variable relationships.

G MUSHROOM

This dataset describes hypothetical samples representing 23 species of gilled mushrooms from the Agaricus and Lepiota Family. Each species is classified as edible or poisonous. As the Guide emphasizes, there is no simple rule for determining mushroom edibility, unlike the straightforward identification of Poisonous Oak and Ivy. This dataset comprises exclusively categorical features, each representing a distinct morphological characteristic of the mushrooms. These features can be used to predict the edibility of the mushroom.

ML Utility Table 14 presents the results for the Mushroom dataset. Despite the lack of straightforward rules for determining edibility, downstream classifiers achieved high performance. TVAE and the CTAB-GAN family of models emerged as the top performers. Notably, TVAE surpassed the original dataset and all other methods across all metrics, demonstrating gains of 0.02 in MCC, ~1 % in macro precision, ~1.2 % in macro recall, and ~1.2 % in macro F-score. DMs and GANs also outperformed the original dataset, albeit with smaller margins. LLMs, however, struggled with this dataset, again underperforming the original dataset across all metrics. These results further reinforce the observation that LLMs, in their current form, may not be the most effective approach for oversampling and addressing class imbalance in downstream tasks.

Fidelity Fig. 18 displays the correlation heatmaps for the Mushroom dataset, revealing interesting insights into the relationship between fidelity and oversampling performance. LLMs and AutoDiffusion demonstrate strong fidelity, closely replicating the correlation structure of the original data. However, this high fidelity does not translate to optimal oversampling performance, suggesting that excessive adherence to the original data distribution may limit the diversity and effectiveness of synthetic samples for data augmentation. TVAE achieves middle-ground performance regarding fidelity, neither perfectly replicating nor significantly deviating from the original correlations. This suggests that a moderate level of fidelity may be more effective

Table 15: MCC, accuracy, precision, recall, and F-Score for the tested methods in the **Oil** dataset.

Model	MCC	Acc.	Precision		Recall		F-Score	
			Weighted	Macro	Weighted	Macro	Weighted	Macro
None	0.34	96.2%	94.8%	75.0%	96.2%	61.2%	95.2%	65.0%
SMOTE [13]	0.39	94.3%	95.3%	74.1%	94.3%	67.7%	94.5%	67.7%
ADASYN [14]	0.39	94.0%	95.3%	73.1%	94.0%	**68.2%**	94.4%	67.9%
TVAE [28]	0.31	90.9%	94.6%	69.2%	90.9%	63.9%	92.2%	63.7%
CTGAN [28]	0.35	95.1%	94.9%	72.0%	95.1%	64.7%	94.9%	66.8%
GaussianCopula [16]	0.35	96.2%	95.0%	75.9%	96.2%	61.9%	95.3%	65.8%
CopulaGAN [28]	0.34	91.7%	95.1%	72.8%	91.7%	64.9%	92.7%	64.1%
CTAB-GAN [29]	-	-	-	-	-	-	-	-
CTAB-GAN+ [30]	-	-	-	-	-	-	-	-
AutoDiffusion [32]	0.39	95.2%	95.1%	71.8%	95.2%	68.1%	95.1%	**69.1%**
ForestDiffusion [31]	**0.40**	**96.3%**	**95.5%**	**80.6%**	**96.3%**	63.3%	**95.5%**	68.0%
GReaT [33]	0.16	77.8%	93.5%	54.6%	77.8%	64.5%	83.7%	52.7%
Tabula [34]	0.17	80.6%	93.6%	55.2%	80.6%	64.6%	85.7%	54.4%

Figure 19: Heatmaps of the pair-wise correlation of the synthetic versus the original data in the **Oil** dataset. Pixels represent the divergence between the synthetic and real feature correlations. Ideally, the synthetic data should have the same correlation between features as the real data. This would be equivalent to a heatmap with all white. Lighter colors indicate that the synthetic data is better at replicating the real data (lighter is better). Methods denoted by a red cross (✗) were unable to generate a sufficient number of samples for the target classes.

for oversampling, allowing for sufficient creativity when generating synthetic samples while still respecting underlying data relationships. Conversely, the poorest performing models in terms of fidelity (CTGAN, CopulaGAN, GaussianCopula, and ForestDiffusion) exhibit significant deviations from the original correlations. This can lead to the generation of unrealistic and potentially misleading synthetic data, hindering the effectiveness of oversampling. However, in this specific case, the impact of poor fidelity on overall performance is less pronounced, likely due to the relatively simple variable relationships within the Mushroom dataset.

H OIL

This dataset, representing the environmental domain, is the most imbalanced in our study. It originates from satellite images of the ocean, categorized into those with and without oil spills. These images were segmented and processed using computer vision algorithms to extract descriptive feature vectors for each image section. The task is to classify image patches as 'Oil Spill' (positive, minority class) or 'Non-spill' (negative, majority class) based on these features. This represents a real-world problem: detecting oil spills, whether from illegal dumping or accidents, which pose significant environmental threats. The high class imbalance, however, presents a challenge for traditional machine learning models, potentially hindering accurate oil spill detection.

ML Utility Table 15 presents the results for the Oil Spill dataset, which, as evidenced by the low MCC scores, poses a significant challenge for ML models. DMs again demonstrate strong performance, outperforming the original dataset in most metrics. Notably, ForestDiffusion achieves a 0.06 improvement in MCC, a ~5.6 % increase in macro precision, a ~2.1 % increase in macro recall, and a ~3 % improvement in macro F-score. While local methods like SMOTE and ADASYN attain the highest macro recall (a ~7 % improvement), AutoDiffusion secures the best macro F-score with a ~4.1 % improvement. Interestingly, the CTAB-GAN family of models encounters difficulties generating samples for this dataset.

Fidelity Fig. 19 displays pair-wise correlation distances, offering insights into the ability of different models to preserve variable relationships in this dataset. LLMs demonstrate strong fidelity, achieving low correlation distances, yet this again does not translate into superior oversampling performance. AEs and DMs also performed well, closely trailing LLMs, presenting good cross-correlations. Conversely, GAN-based methods struggled to reproduce variable relationships accurately (or in some instances not being able to generate correct samples), potentially hindering their effectiveness, which, in this case, negatively impacts oversampling performance. As previously noted, while some deviation from the original data distribution can be beneficial for generating diverse synthetic samples, a failure to reasonably reproduce correlations can hinder the effectiveness of oversampling.