# Tabular and Feature Space Synthetic Data Generation: A Literature Review

Joao Fonseca[1*], Fernando Bacao[1]

[1]NOVA Information Management School, Universidade Nova de Lisboa

*Corresponding Author

Postal Address: NOVA Information Management School, Campus de Campolide, 1070–312 Lisboa, Portugal

Telephone: +351 21 382 8610

The generation of synthetic data can be used for anonymization, regularization, oversampling, semi-supervised learning, self-supervised learning and various other tasks. The wide range of applications of these mechanisms motivated the development of new algorithms specialized in generating data for specific types of data and Machine Learning (ML) tasks. As a result, the analysis of the different types of generative models

## 1 Introduction

Synthetic data is obtained from a generative process based on properties of real data [1]. The generation of synthetic data is essential for various domains and tasks. For example, synthetic data is used as a form of regularizing neural networks (*i.e.*, data augmentation) **[CITATION]**. One form of anonymizing datasets is via the production of synthetic observations (*i.e.*, synthetic data generation) **[CITATION]**. In settings where only a small portion of training data is labeled, some techniques generate artificial data using both labeled and unlabeled data with a modified loss function to train neural networks (*i.e.*, semi-supervised learning) [2]. In imbalanced learning contexts, synthetic data can be used to balance the target classes' frequencies and reinforce the learning of minority classes (*i.e.*, oversampling) [3]. Some active learning frameworks use data generation to improve the quality of data selection and classifier training [4]. Other techniques employ data generation to produce deep neural networks without labeled data (*i.e.*, self-supervised learning) [5].

The breadth of these techniques span multiple domains, such as facial recognition [6], Land Use/Land Cover mapping **[CITATION]**, medical image processing **[CITATION]**, Natural Language Processing (NLP) [7] or credit card default prediction [8]. According to the domain and data type, the data generation techniques used may vary significantly. Generally speaking, some data generation mechanisms are specific to some domains, data types or tasks. For example, . . . . Most, if not all, of these techniques are applied on the input or output space.

However, there are various data generation techniques that are invariant to the task or data types used. These techniques can be either applied in the feature space [9] or in tabular datasets[1]. On one hand,

[1] Tabular data is a database structured in tabular form, composed of columns (features) and rows (observations) [10]

data generation in the feature space uses a generative model to learn a manifold, lower-dimensional abstraction over the input space [11], defined here as the feature space. At this level, any tabular data generation mechanism can be applied and reconstructed into the input space if necessary. On the other hand, synthetic data generation on tabular data can be applied to most problems. Although, the choice of generation mechanism is still dependant on (1) the importance of the relationships found between the different features, (2) the ML task developed and (3) the motivation for the generation of synthetic data. For example, when generating data to address an imbalanced learning problem (*i.e.*, oversampling), the relationships between the different features are not necessarily kept since the goal is to reinforce the learning of the minority class by redefining an ML classifier's decision boundaries. If the goal is to anonymize a dataset, perform some type of descriptive task, or ensure a consistent model interpretability, these relationships need to be kept.

Depending on the context, evaluating the quality of the generated data is a complex task. For example, for image and time series data, perceptually small changes in the original data can lead to large changes in the euclidean distance [1, 12]. The evaluation of generative models typically account primarily for the performance in a specific task, since good performance in one criterion does not imply good performance on another [12]. However, in computationally intensive tasks it is often impracticable to search for the optimal configurations of generative models. To address this limitation, other evaluation methods have been proposed to assist in this evaluation, which can be distinguished into statistical divergence metrics and precision/recall metrics [13]. The relevant performance metrics found in the literature are discussed in Section 6.

## 1.1 Motivation, Scope and Contributions

This literature review focuses on generation mechanisms applied to tabular data and the different ML techniques where tabular synthetic data is used. In addition, we focus on the ML perspective of synthetic data, as opposed to the practical perspective. From a practical sense, synthetic data is used as a proxy of real data. It is assumed to be inaccessible, essential and a secondary asset for tasks like education, software development, or systems demonstrations [14].

We focus on data generation techniques in the tabular and feature space (*i.e.*, embedded inputs) with a focus on classification and associated ML problems. Related literature reviews are mostly focused on specific algorithmic or domain applications, with little to no emphasis on the core generative process. For this reason, these techniques often appear "sandboxed", even though there is a significant overlap between them. There are some related reviews published since 2019. Assefa et al. [1] provides a general overview of synthetic data generation for time series data anonymization in the finance sector. Hernandez et al. [15] reviews data generation techniques for tabular health records anonymization. Raghunathan [16] reviews synthetic data anonymization techniques that preserve the statistical properties of a dataset. Nalepa et al. [17] reviews data augmentation techniques for brain-tumor segmentation. Bayer et al. [18] distinguishes augmentation techniques for text classification into feature and data space, while providing an extensive overview of augmentation methods within this domain. However, the taxonomy proposed and feature space augmentation methods are not necessarily specific to the domain. Shorten et al. [19], Chen et al. [20], Feng et al. [7] and Liu et al. [21] also review data augmentation techniques for text data. Yi et al. [22] review Generative Adversarial Network architectures for medical imaging. Wang et al. [23] reviews face data augmentation techniques. Shorten et al. [24] and Khosla et al. [25] discuss techniques for image data augmentation. Iwana et al. [26] and Wen et al. [27] also review time series data augmentation techniques. Zhao et al. [28] review data augmentation techniques for graph data. The analysis of related literature reviews [2] is shown in Table 1.

[2] published since 2019, using the search query (`"synthetic data generation" OR "oversampling" OR "im-`

Table 1: Related literature reviews published since 2019.

| Reference | Data type | ML problem | Domain | Observations |
|---|---|---|---|---|
| Assefa et al. [1] | — | Differential privacy | Finance | Analysis of applications, motivation and properties of synthetic data for anonymization. |
| Hernandez et al. [15] | Tabular | Differential privacy | Healthcare | Focus on GANs. |
| Raghunathan [16] | Tabular | Differential privacy | Statistics | Focus on general definitions such as differential privacy and statistical disclosure control. |
| Nalepa et al. [17] | Image | Segmentation | Medicine | Analysis of algorithmic applications on a 2018 brain-tumor segmentation challenge. |
| Bayer et al. [18] | Text | Classification | — | Distinguish 100 methods into 12 groups. |
| Shorten et al. [19] | Text | Deep Learning | — | General overview of text data augmentation. |
| Chen et al. [20] | Text | Few-shot Learning | — | Augmentation techniques for machine learning with limited data |
| Feng et al. [7] | Text | — | — | Overview of augmentation techniques and applications on NLP tasks. |
| Liu et al. [21] | Text | — | Various | Analysis of industry use cases of data augmentation in NLP. Emphasis on input level data augmentation. |
| Yi et al. [22] | Image | — | Medicine | Emphasis on GANs. |
| Wang et al. [23] | Image | Deep Learning | — | Regularization techniques using facial image data. Emphasis on Deep Learning generative models. |
| Shorten et al. [24] | Image | Deep Learning | — | Emphasis on data augmentation as a regularization technique. |
| Khosla et al. [25] | Image | — | — | Broad overview of image data augmentation. Emphasis on traditional approaches. |
| Iwana et al. [26] | Time series | Classification | — | Defined a taxonomy for time series data augmentation. |
| Wen et al. [27] | Time series | Various | — | Analysis of data augmentation methods for classification, anomaly detection and forecasting. |
| Zhao et al. [28] | Graph | Various | — | Graph data augmentation for supervised and self-supervised learning. |
| Khalifa et al. [29] | Image | — | Various | General overview of image data augmentation and relevant domains of application. |

The different taxonomies established in the literature follow a similar philosophy, but vary in terminology and are often specific to the technique discussed. Regardless, it is possible to establish a broader taxonomy without giving up on specificity. This study provides a joint overview of the different data generation approaches, domains and ML techniques where data generation is being used, as well as a common taxonomy across domains. It extends the analyses found in these articles and uses the compiled knowledge to identify research gaps. We compare the strengths and weaknesses of the models developed within each of these fields. Finally, we identify possible future research directions to address some of the limitations found. The contributions of this paper are summarized below:

`balanced learning" OR "data augmentation") AND ("literature review" OR "survey").` Retrieved on August 11$^{th}$, 2022. More articles were added later whenever found relevant.

- Bridge different ML concepts using synthetic data generation in its core (Algorithmic applications + Review of the State-of-the-art).
- Propose a synthetic data generation/data augmentation taxonomy to resolve the ambiguity in the literature (Data augmentation taxonomy).
- Characterize all relevant data generation methods using the proposed taxonomy.
- Discuss the ML techniques in which synthetic data generation/data augmentation is used, beyond regularization and consolidate the current data generation mechanisms across the different techniques (Algorithmic Applications).
- Bring to light the key challenges of synthetic data generation and put forward possible research directions in the future.

### 1.2 Paper Organization

This paper is organized as follows: Section 2 defines and formalizes the different concepts, goals, trade-offs and motivations related to synthetic data generation. Section 3 establishes the taxonomy used to categorize all the methods described in the paper. Section **??** reviews synthetic data generation mechanisms in the feature space. Section **??** reviews synthetic data generation mechanisms in the input space. Section 4 describes the applications of synthetic data in ML methods. Section 6 reviews performance evaluation methods of synthetic data generation mechanisms. Section 7 summarizes the main findings and discusses limitations and possible research directions in the state-of-the-art. Section 9 presents the main conclusions drawn from this study.

## 2 Background

In this section we define basics concepts, common goals, trade-offs and motivations regarding the generation of synthetic data in ML. We define synthetic data generation as the production of observations using a generative model (regardless of its nature) that resemble naturally occurring observations within a certain domain. It requires access to either a training dataset, a generative process, or a data stream. However, additional requirements might be imposed depending on the ML task being developed. For example, to generate artificial data for regularization purposes in supervised learning (*i.e.*, data augmentation) the training dataset must be annotated **[CITATION]**. The generation of synthetic data for anonymization purposes assumes synthetic datasets to be different from the original data, while following the same statistical properties **[CITATION]**. Domain knowledge may also be necessary to encode specific relationships among features into the generative process.

### 2.1 Relevant Learning Problems

The breach of sensitive information is an important barrier to the sharing of datasets, especially when it concerns personal information [30]. A common solution for this problem is the generation of synthetic data without identifiable information. Generally speaking, ML tasks that require data with sensitive information are not compromised when using synthetic data. The experiment conducted by Patki et al.

[31] using relational datasets showed that in 11 out 15 comparisons ($\approx 73\%$), practitioners performing predictive modelling tasks using fully synthetic datasets performed the same or better than those using the original dataset. This topic is discussed in Section 4.1.

A common problem in the training of deep neural networks are their capacity to generalize [32] (*i.e.*, reduce the difference in classification performance between known and unseen observations). Data augmentation is a common method to address this problem. The generation of synthetic observations increases the range of the possible input space used in the training phase, which reduces the performance difference between known and unseen observations. Although other regularization methods exist, data augmentation is a useful method since it does not affect the choice in the architecture of the ML classifier and does not exclude the usage of other regularization methods. In domains such as computer vision and NLP, data augmentation is also used to improve the robustness of models against adversarial attacks [33, 34]. These topics are discussed into higher detail in Section 4.2.

In supervised learning, synthetic data generation is often motivated by the need to balance target class distributions (*i.e.*, oversampling). Since most ML classifiers are designed to perform best with balanced datasets, defining an appropriate decision boundary to distinguish rare classes becomes difficult [35]. Although there are other approaches to address imbalanced learning, oversampling techniques are generally easier to implement since they do not involve modifications to the classifier. This topic is discussed into higher detail in Section 4.3.

In supervised learning projects where labeled data is not readily available, but can be labeled, an Active Learning (AL) method may be used to improve the labelling process. AL aims to reduce the cost of producing training datasets by finding the most informative observations to label and feed into the classifier [36]. In this case, the generation of synthetic data is particularly useful to reduce the amount of labelled data required for a successful ML project and its costs. A similar motivation applies to the case of few-shot learning: small datasets may be expanded with synthetic data [37]. These topics are discussed in Sections 4.4 and **??**.

The two other techniques reliant on synthetic data generation is Semi-supervised and Self-supervised learning. The former leverages both labeled and unlabeled data in the training phase, simultaneously. Most of the methods in the literature apply perturbations on the training data as part of the training procedure [38]. Self-supervised learning is a technique used to train neural networks in the absence of labeled data. Both techniques use synthetic data generation as an internal procedure for most of these methods. These techniques are discussed in Sections 4.5 and 4.6.

## 2.2 Problem Formulation

The original dataset, $\mathcal{D} = \mathcal{D}_L \cup \mathcal{D}_U$, is a collection of real observations and is distinguished according to whether a target feature exists, $\mathcal{D}_L = ((x_i, y_i))_{i=1}^{l}$, or not, $\mathcal{D}_U = (x_i)_{i=1}^{u}$. All three datasets, $\mathcal{D}$, $\mathcal{D}_L$ and $\mathcal{D}_U$ consist of ordered collections with lengths $l + u$, $l$ and $u$, respectively. Synthetic data generation is performed using a generator, $f_{gen}(x; \tau) = \tilde{x}$, where $\tau$ defines the generation policy (*i.e.*, its hyperparameters), $x \in \mathcal{D}$ is an observation and $\tilde{x} \in \mathcal{D}^s$ is a synthetic observation. Analogous to $\mathcal{D}$, the synthetic dataset, $\mathcal{D}^s$, is also distinguished according to whether there is an assignment of a target feature, $\mathcal{D}^s_L = ((\tilde{x}_j, \tilde{y}_j))_{j=1}^{l'}$, or not, $\mathcal{D}^s_U = (\tilde{x}_j)_{j=1}^{u'}$.

Depending on the ML task, it may be relevant to establish metrics to measure the quality of $\mathcal{D}^s$. In this case, a metric $f_{qual}(\mathcal{D}^s, \mathcal{D})$ is used to determine the level of similarity/dissimilarity between $\mathcal{D}$ and $\mathcal{D}^s$. In addition, a performance metric to estimate the performance of a model on the objective task, $f_{per}$, may be used to determine the appropriateness of a model with parameters $\theta$, *i.e.*, $f_\theta$. The generator's goal is to generate $\mathcal{D}^s$ with arbitrary length, given $\mathcal{D} \sim \mathbb{P}$ and $\mathcal{D}^s \sim \mathbb{P}^s$, such that $\mathbb{P}^s \approx \mathbb{P}$, $x_i \neq x_j \forall x_i \in \mathcal{D} \wedge x_j \in \mathcal{D}^s$. $f_{gen}(x;\tau)$ attempts to generate a $\mathcal{D}^s$ that maximizes either $f_{per}$, $f_{qual}$, or a combination of both.

## 3 Data Generation Taxonomy

The taxonomy proposed in this paper is a compilation of different definitions found in the literature, along with other traits that vary among domains and generation techniques. Within image data studies, Shorten et al. [24] and Khalifa et al. [29] divide data augmentation techniques into "basic" or "classical" approaches and deep learning approaches. In both cases, the former refers to domain-specific generation techniques, while the latter may be applied to any type of data. Iwana et al. [26] proposes a time-series data augmentation taxonomy divided in four families: (1) Decomposition, (2) Pattern mixing, (3) Generative models and (4) Decomposition. With exception to generative models, the majority of the methods presented in the remaining families are well established and domain specific. Hernandez et al. [15] defines a taxonomy for synthetic tabular data generation approaches divided in three types of approaches: (1) Classical, (2) Deep learning and (3) Others. Most taxonomies found followed similar definitions with variations in terminology or distinction criteria. In addition, all taxonomies with categories defined as "basic", "traditional" or "classical" use these to characterize domain-specific transformations.

Within the taxonomies found, none of them consider how a generation mechanism employs $\mathcal{D}$ into the generation process or, if applicable, the training phase. However, it is important to understand whether a generation mechanism randomly selects $x$ and a set of close neighbors, thus considering local information only, or considers the overall dataset or data distribution for the selection of $x$ and/or generation of $\tilde{x}$. Our proposed taxonomy is depicted in Figure 1. It characterizes data generation mechanisms using four properties:

1. Architecture. Defines the broader type of data augmentation. It is based on domain specificity, architecture type or data transformations using a heuristic or random perturbation process. Generation techniques that apply a form of random perturbation, interpolation or geometric transformation to the data with some degree of randomness are considered randomized approaches. Typical, domain-specific data generation techniques are considered traditional architectures. These techniques apply transformations to a data point using *a priori* domain knowledge. Generative models based on neural network architectures are defined as network-based. These architectures attempt to either generate observations in the feature space and/or by producing observations that are difficult to distinguish from the original dataset.

2. Application level. Refers to the phase of the ML pipeline where the generative process is included. Generative models are considered internal if they are used alongside the primary ML task, whereas models used prior to the development of the primary ML task are considered external.

3. Scope. Considers the usage of the original dataset's properties. Generative models that consider the density of the data space, statistical properties of $\mathcal{D}$, or attempt to replicate specific relationships found in $\mathcal{D}$ are considered to have a global scope, whereas generative models that consider a single observation and/or a set of close neighbors are considered to have a local scope. On the one hand, generative models with a local scope do not account for $\mathbb{P}^s$ but allow for a larger diversity of candidate $x^s$ and higher variance within $\mathcal{D}^s$. On the other hand, generative models with a global scope have a higher capacity to model $\mathbb{P}^s$ but produce candidate $x^s$ with lower diversity and lower variance within $\mathcal{D}^s$.

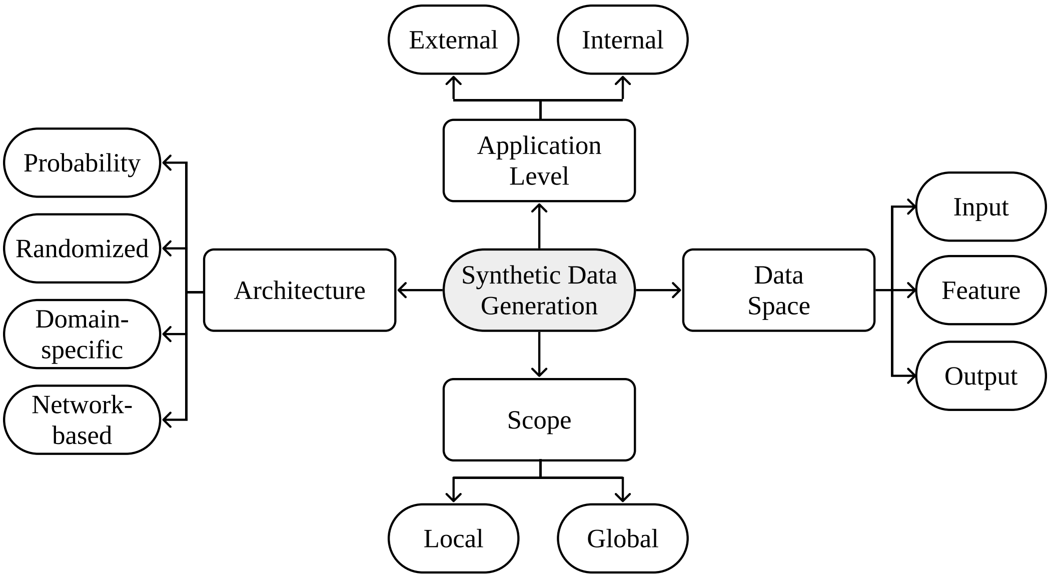


Figure 1: General taxonomy of data generation mechanisms proposed in this paper.

4. Data space. Refers to the type data representation used to apply the generative model. Generation mechanisms can be applied using the raw dataset (*i.e.*, on the input space), an embedded representation of the data (*i.e.*, on the feature space) or based on the target feature (*i.e.*, on the output space). Although some studies discuss the need to generate synthetic data on the input space [30, 31], there are various studies that apply synthetic data generation techniques on a feature space.

Throughout the analysis of the different types of generation mechanisms, all relevant methods were characterized using this taxonomy and listed in Table 2.

Table 2: Summary of the synthetic data generation methods discussed in this work.

| Algorithm | ML Problem | Type | Architecture | Level | Data Space | Scope |
|---|---|---|---|---|---|---|
| SDV [31] | Anon. | PDF | Probabilistic | External | Input | Global |
| MST [39] | DP | Marginal | Probabilistic | External | Input | Global |
| MWEM [40] | DP | Marginal | Probabilistic | External | Input | Global |
| MWEM-PGM [41] | DP | PGM | Probabilistic | External | Input | Global |
| PrivBayes [42] | DP | PGM | Probabilistic | External | Input | Global |
| DPGAN [43] | DP | GAN | Network | External | Feature | Global |
| DPCTGAN [44] | DP | GAN | Network | External | Feature | Global |
| PATE-GAN [45] | DP | GAN | Network | External | Feat. + Out. | Global |
| PATECTGAN [44] | DP | GAN | Network | External | Feat. + Out. | Global |
| FEM [46] | DP | Workload | Probabilistic | External | Input | Global |
| RAP [47] | DP | Workload | Probabilistic | External | Input | Global |
| PDF [48, 49] | — |  | Probabilistic | External | Input | Global |
| Kamino [50] | DP |  | Probabilistic | External | Input | Global |
| RON-GAUSS [51] | DP | PDF | Probabilistic | Internal | Feature | Global |
| HDMM [52] | DP |  | Probabilistic | External | Input | Global |
| DualQuery [53] | DP |  | Probabilistic | External | Input | Global |
| ROS(E) [54] | Ovs | Bootstrap | Randomized | External | Input | Local |
| SMOTE [55] | Ovs | Linear | Randomized | External | Input | Local |
| SMOTENC [55] | Ovs | Linear | Randomized | External | Input | Local |

Continued on next page

Table 2: Summary of the synthetic data generation methods discussed in this work.

| Algorithm | ML Problem | Type | Architecture | Level | Data Space | Scope |
|---:|:---:|:---:|:---:|:---:|:---:|:---:|
| SMOTEN [55] | Ovs | Linear | — | External | Input | Local |
| Borderline-SMOTE [56] | Ovs | Linear | Randomized | External | Input | Local |
| G-SMOTE [57] | Ovs | Geometric | Randomized | External | Input | Local |
| ADASYN [58] | Ovs | Linear | Randomized | External | Input | Local |
| KernelADASYN [59] | Ovs | PDF | Probabilistic | External | Input | Local |
| MOKAS [60] | Ovs | Rec. Err. | Network | External | Feature | Global |
| SOMO [61] | Ovs | Linear | Net.+Rand. | External | Input | Global |
| G-SOMO [62] | Ovs | Geometric | Net.+Rand. | External | Input | Global |
| VAE [63] | Ovs | | | | | |
| GMM [64] | | | | | | |
| GMM [65] | | | | | | |
| Safe-level SMOTE [66] | Ovs | Linear | Randomized | External | Input | Local |
| LR-SMOTE [67] | Ovs | Linear | Randomized | External | Input | Global |
| K-means SMOTE [68] | Ovs | Linear | Randomized | External | Input | Global |
| DBSMOTE [69] | Ovs | Linear | Randomized | External | Input | Local |
| CGAN [70] | Ovs | GAN | Network | External | Feature | Global |
| K-means CTGAN [71] | Ovs | GAN | Network | External | Feature | Global |
| SMOTER [72] | Ovs + Reg | Linear | Randomized | External | Input | Local |
| G-SMOTER [73] | Ovs + Reg | Linear | Randomized | External | Input | Local |
| RACOG [74] | Ovs | PGM | Probabilistic | External | Input | Global |
| wRACOG [74] | Ovs | PGM | Probabilistic | External | Input | Global |
| RWO [75] | Ovs | RW | Probabilistic | External | Input | Global |
| PDFOS [76] | Ovs | PDF | Probabilistic | External | Input | Global |
| Mixup [77] | DA | Linear | Randomized | External | In.+Out. | Local |
| M-Mixup [78] | DA | Linear | Network | Internal | Feat.+Out. | Global |
| NL-Mixup [79] | DA | Geometric | Randomized | External | In.+Out. | Local |
| AE-DA [80] | DA | AE | Network | External | In./Feat.+Out. | Local |
| MODALS [81] | DA | — | Network | Internal | Feat. | Global |
| LSI [82] | DA | AE | Network | External | Feat.+Out. | Global |
| Gibbs [83] | DA | PGM | Probabilistic | External | Input | Global |
| MedGAN [84] | DA | GAN | Network | External | Feature | Global |
| GANBLR [85] | DA | PGM | Probabilistic | External | Input | Global |
| Table-GAN [86] | DA | GAN | Network | External | Feature | Global |
| CTGAN [87] | DA | GAN | Network | External | Feature | Global |
| TVAE [87] | DA | AE | Network | External | Feature | Global |
| AE [88] | DA | AE | Network | External | Feature | Global |
| InfoMixup [4] | AL | Linear | Network | Internal | Feat.+Out. | Global |
| VAEACGAN [89] | AL | AE | Network | Internal | Feature | Global |
| AL-G-SMOTE [36] | AL | Geometric | Randomized | Internal | Input | Local |
| DAE [90] | Semi-SL | AE | Network | Internal | Input | Global |
| Π-model [91] | Semi-SL | PDF | Randomized | Internal | In.+Feat. | Local |
| Mean Teacher [92] | Semi-SL | PDF | Randomized | Internal | In.+Feat. | Local |
| ICT [93] | Semi-SL | Linear | Randomized | Internal | Input | Local |
| Mixmatch [94] | Semi-SL | Linear | Randomized | Internal | Input | Local |
| SDAT [95] | Semi-SL | AE+PDF | Net.+Prob. | Internal | Feature | Global |
| MCoM [96] | Semi-SL | Linear | Randomized | Int.+Ext. | Inp.+Feat. | Global |
| C-Mixup [97] | Semi/Self-SL | AE+Lin. | Net+Rand. | Internal | Feature | Global |
| VIME [10] | Semi/Self-SL | Mask | Randomized | Internal | Input | Local |
| SubTab [98] | Self-SL | Mask | Rand.+Prob. | Internal | Input | Local |
| Scarf [99] | Self-SL | Mask | Randomized | Internal | Input | Local |
| A-SFS [100] | Self-SL | Mask | Randomized | Internal | Input | Local |

# 4 Algorithmic applications

In this section we discuss the data generation mechanisms for the different contexts where they are applied. We emphasize the constraints in each problem that condition the way generation mechanisms are used.

## 4.1 Privacy

Synthetic data generation is a technique used to produce synthetic, anonymized versions of datasets [30]. It is considered a good approach to share sensitive data without compromising significantly a given data mining task [101, 86]. Traditional data anonymization techniques, as well as federated learning are two other viable solutions for privacy-preserving data publishing tasks, but contain drawbacks [15]. On the one hand, traditional data anonymization requires domain knowledge, is labor intensive and remains susceptible to disclosure [102]. On the other hand, federated learning is a technically complex task that consists on training ML classifiers on edge devices and aggregating temporarily updated parameters on a centralized server, instead of aggregating the training data [103]. Although it prevents sharing sensitive data, its applicability is dependent on the task. Dataset anonymization via synthetic data generation attempts to balance disclosure risk and data utility in the final synthetic dataset. The goal is to ensure observations are not identifiable and the relevant data mining tasks are not compromised [104, 105].

The generation of synthetic datasets allow a more flexible approach to the successful implementation of ML tasks. To do this, it is important to guarantee that sensitive information in $\mathcal{D}$ is not leaked into $\mathcal{D}^s$. Differential privacy (DP), a formalization of privacy, offers strict theoretical privacy guarantees [44]. A differentially private generation mechanism produces a synthetic dataset, regulated by the privacy parameter $\epsilon$, with statistically indistinguishable results when using either $\mathcal{D}$ or neighboring datasets $\mathcal{D}' = \mathcal{D} \backslash \{x\}$, for any $x \in \mathcal{D}$. A synthetic data generation model ($f_{gen}$) guarantees $(\epsilon, \delta)$-differential privacy if $\forall S \subseteq Range(f_{gen})$ all $\mathcal{D}, \mathcal{D}'$ differing on a single entry [40]:

$$Pr[f_{gen}(\mathcal{D}) \in S] \leq e^{\epsilon} \cdot Pr[f_{gen}(\mathcal{D}') \in S] + \delta \tag{1}$$

In this case, $\epsilon$ is a non-negative number defined as the privacy budget. A lower $\epsilon$ guarantees a higher level of privacy, but reduces the quality of the produced synthetic data. The generation of DP synthetic data is especially appealing since DP is not affected by post-processing; any ML pipeline may be applied using $\mathcal{D}^s$ without losing differential privacy [106].

Despite the formalization and the ability to quantify differential privacy, there are popular synthetic data-based anonymization approaches that perform this task without DP guarantees. Specifically, the Synthetic Data Vault (SDV) [31] is a method for database anonymization that uses Gaussian Copula models for generating data. However, this method allows the usage of other generation mechanisms. A posterior extension of SDV was proposed to generate data using a CTGAN [87] and to handle sequential tabular data using a conditional probabilistic auto-regressive neural network [107].

The choice of the most appropriate DP synthetic data generation techniques depends on the task to be developed (if known) and the domain. However, marginal-based algorithms appear to perform well across various tests [108]. A well-known method for the generation of DP synthetic datasets is the combination of the Multiplicative Weights update rule with the Exponential Mechanism (MWEM) [40]. The MWEM mechanism is an active learning-style algorithm that maintains an approximation of $\mathcal{D}^s$. At each time step, MWEM selects the worst approximated query (determined by a scoring function) using the Exponential

Mechanism and improves the accuracy of the approximating distribution using the Multiplicative Weights update rule. A know limitation of this method refers to its scalability. Since this method represents the approximate data distribution in datacubes, this method becomes infeasible for high-dimensional problems [41]. This limitation was addressed with the integration of a Probabilistic Graphical Model-based (PGM) estimation into MWEM (MWEM-PGM) and a subroutine to compute and optimize the clique marginals of the PGM, along with other existing privacy mechanisms [41]. Besides MWEM, this method was used to modify and improve the quality of other DP algorithms: PrivBayes [42], HDMM [52] and DualQuery [53].

PrivBayes [42] circumvents the curse of dimensionality by computing a differentially private Bayesian Network (*i.e.*, a type of PGM). Instead of injecting noise into the dataset, they inject noise into the lower-dimensional marginals. The high-dimensional matrix mechanism (HDMM) [52] mechanism is designed to efficiently answer a set of linear queries on high-dimensional data, which are answered using the Laplace mechanism. The DualQuery algorithm [53] is based on the two-player interactions in MWEM, and follows a similar synthetic data generation mechanism as the one found in MWEM.

FEM [46] follows a similar data generation approach as MWEM. It also uses the exponential mechanism and replaces the multiplicative weights update rule with the follow-the-perturbed-leader (FTPL) algorithm [109]. The Relaxed Adaptive Projection (RAP) algorithm [47] uses the projection mechanism [110] to answer queries on the private dataset using a perturbation mechanism and attempts to find the synthetic dataset that matches the noisy answers as accurately as it can.

Kamino [50] introduces denial constraints in the data synthesis process. Kamino builds on top of the probabilistic database framework (PDF) [48, 49], which uses ordinary databases to model a probability distribution and integrates denial constraints as parametric factors, out of which the synthetic observations are sampled. RON-GAUSS [51] combines the random orthonormal (RON) dimensionality reduction technique and synthetic data sampling using either a Gaussian generative model or a Gaussian mixture model. The motivation for this model stems from the *Diaconis-Freedman-Meckes* effect [111], which states that most high-dimensional data projections follow a nearly Gaussian distribution. Since RON-GAUSS includes a feature extraction step (using RON) and the synthetic data generated is not projected back into the input space, we consider RON-GAUSS an internal approach to the ML pipeline.

The MST mechanism [39] is a marginal estimation-based approach that produces differentially private data. It uses the Private-PGM mechanism [41] that relies on the PGM approach to generate synthetic data. PGM models are most commonly used when it is important to maintain the pre-existing statistical properties and relationships between features [112].

Another family of DP synthetic data generation techniques relies on the usage of Generative Adversarial Networks (GAN). DPGAN [43] modifies the original GAN architecture to make it differentially private by introducing noise to gradients during the learning procedure. This approach was also applied on a conditional GAN architecture directed towards tabular data (CTGAN) [87], which originated the DPCTGAN [44]. Another type of GAN-based DP data synthesis method is based on the combination of a GAN architecture and the Private Aggregation of Teacher Ensembles (PATE) [113] approach. Although the PATE method generates a DP classifier, it served as the basis for PATE-GAN [45], a DP synthetic data generation mechanism. PATE-GAN replaces the discriminator component of a GAN with the PATE mechanism, which guarantees DP over the generated data. The PATE mechanism is used in the learning phase to train an ensemble of classifiers to distinguish real from synthetic data. In a second step, the predicted labels are passed (with added noise) to another discriminator, which is used to train the generator network.

## 4.2 Regularization

When there are no underlying problems in the training data, it is sampled from a fixed data source, is labeled, and balanced, the resulting ML classifier is expected to achieve good generalization performance [114]. However, if one or more of these assumptions does not hold, the ML model becomes prone to overfitting [115]. Regularization techniques are often used to address problems like overfitting, small training dataset, high dimensionality, outliers, label noise and catastrophic forgetting [116, 117, 118, 119]. They can be divided into three groups [120]:

1. Output level modifications. Transforms the labels in the training data.

2. Algorithmic level modifications. Modifies the classifier's architecture, loss function or other components in the training procedure.

3. Input level modifications. Modifies the training dataset by expanding it with synthetic data.

The last approach, input level modifications, is known as data augmentation. It is used to increase the size and data variability of data in a training dataset, by producing synthetic observations [121, 122]. Since it is applied at the data level, it can be used for various types of problems and classifiers [123]. Earlier definitions of data augmentation refer to methods based on iterative optimization or sampling algorithms that introduce unobserved data or latent variables [124]. In the current ML literature, data augmentation techniques mostly refer to the former, while the latter is better known as feature extraction. Although data augmentation is commonly used and extensively studied in computer vision [24] and natural language processing [7], research on tabular data augmentation is sparse.

Mixup [77] consists of a linear interpolation between two randomly selected observations and their target feature values, $(x_i, y_i), (x_j, y_j) \in \mathcal{D}_L$, such that given $\lambda \sim \text{Beta}(\alpha, \alpha)$, $x^s = \lambda x_i + (1-\lambda)x_j$ and $y^s = \lambda y_i + (1-\lambda)y_j$, where $\alpha$ is a predetermined hyperparameter. This method was the source to Manifold Mixup (M-Mixup) [78]. It generates synthetic data in the feature spaces of a neural network classifier's hidden layers. Another Mixup-based data augmentation approach, Nonlinear Mixup (NL-Mixup) [79], applies a nonlinear interpolation policy. In this case, $\Lambda$ is a set of mixing policies sampled from a beta distribution applied to each feature. This approach modifies the original mixup approach to generate data within a hyperrectangle/orthotope: $x^s = \Lambda \odot x_i + (1-\Lambda) \odot x_j$, where $\odot$ denotes the Hadamard product.

Feng et al. [80] proposed an autoencoder-based data augmentation (AE-DA) approach where the training of the autoencoder is done for each target class, non-iteratively, which reduces the amount of time required compared to the batch processing approach. The decoding weights of an autoencoder is scaled and linearly combined with an observation from another class using a coefficient that follows the beta distribution. The latter step varies from typical interpolation-based approaches, since this coefficient is usually drawn from a uniform distribution.

The Modality-Agnostic Automated Data Augmentation in the Latent Space model (MODALS) [81] leverages on the concept discussed by DeVries et al. [9], as well as the Latent Space Interpolation method (LSI) [82] and M-Mixup [78]. However, MODALS introduces a framework for data augmentation internally. It contains a feature extraction step, trained using a combination of adversarial loss, classification loss and triplet loss, where latent space generation mechanisms are applied. The classifier is trained using the original and the synthetic observations generated in the feature space. In this study the authors discuss Difference transform augmentation method. It generates within-class synthetic data by selecting a $x^c$ and two random observations within the same class, $x_i, x_j$, to compute $x^s = x^c + \lambda(x_i - x_j)$. In addition they also experiment with Gaussian noise and Hard example extrapolation, determined by $x^s = x^c + \lambda(x^c - \mu)$, where $\mu$ is the mean of the observations within a given class.

In the model distillation approach proposed in [83] the student model is trained with synthetic data generated with Gibbs sampling. Although Gibbs sampling is infrequently used in recent literature, two oversampling methods using Gibbs sampling appear to achieve state-of-the-art performance [74]. However, probabilistic-based approaches for data augmentation are uncommon; there are some methods proposed for the more specific case of oversampling, but no more related methods for data augmentation were found.

A well-known approach to GAN-based data augmentation is Table-GAN [86]. It utilizes the vanilla GAN approach to the generation of synthetic data. However, vanilla GAN does not allow the controlled generation of synthetic data given conditional attributes such as the target feature values in supervised learning tasks and may be the cause for aggravated categorical feature imbalance. These limitations were addressed with the CTGAN [87] algorithm, which implements the conditional GAN approach to tabular data. Another GAN-based architecture, MedGAN [84], can also be adapted for tabular data and is used as a benchmark in related studies (*e.g.*, [87, 85]). When compared to the remaining GAN-based approaches, MedGAN's architecture is more complex and is generally outperformed in the experiments reported in the literature. The GANBLR [85] modifies vanilla GAN architectures with a Bayesian network as both generator and discriminator to create synthetic data that is expected to be indistinguishable from real data. This approach benefits from its interpretability and reduced complexity, while maintaining state-of-the-art performance across various evaluation criteria.

Another less popular approach for network-based synthetic data generation are autoencoder architectures. TVAE, proposed in [87] achieved state-of-the art performance. It consists of the VAE algorithm with an architecture modified for tabular data (*i.e.*, 1-dimensional). However, as discussed by the authors, this method contains limitations since it is difficult to achieve DP with AE-based models since they access the original data during the training procedure, unlike GANs. Delgado et al. [88] studies the impact of data augmentation on supervised learning with small datasets. The authors compare four different AE architectures: Undercomplete, Sparse, Deep and Variational AE. Although any of the tested AE architectures improved classification performance, the deep and variational autoencoders were the best overall performing models.

## 4.3 Oversampling

One problem frequently found in industry settings is the training of ML models on imbalanced datasets. Since most supervised machine learning classifiers are designed to expect classes with similar frequencies, with highly skewed distributions in $\mathcal{D}_L$, the classifier's predictions tend to be biased towards overrepresented classes [3]. For example, one can predict correctly with over 99% accuracy whether credit card accounts were defrauded using a constant classifier. This issue can be addressed in 3 different ways: resampling, algorithmic modifications and cost-sensitive solutions [125]. Resampling techniques are more general approaches when opposed to algorithmic and cost-sensitive methods. They modify $\mathcal{D}_L$ to ensure balanced class frequencies by removing majority class observations (*i.e.*, undersampling), producing synthetic minority class observations (*i.e.*, oversampling), or a combination of both. However, since undersampling removes observations from $\mathcal{D}_L$, it has the disadvantage of information loss [126] and lacks effectiveness when compared to oversampling methods [127, 128]. Oversampling can be considered a specific setting of data augmentation.

Oversampling is an appropriate technique when, given a set of $n$ target classes, there is a collection $C_{maj}$

containing the majority class observations and $C_{min}$ containing the minority class observations such that $\mathcal{D}_L = \bigcup_{i=1}^{n} C_i$. The training dataset $\mathcal{D}_L$ is considered imbalanced if $|C_{maj}| > |C_{min}|$. This imbalance is quantified using the Imbalance Ratio (IR), expressed as $IR = \frac{|C_{maj}|}{|C_{min}|}$. An oversampling algorithm with a standard generation policy will generate a $\mathcal{D}_L^s = \bigcup_{i=1}^{n} C_i^s$ that guarantees $|C_i \cup C_i^s| = |C_{maj}|, \forall i \in \{1, \dots, n\}$. The model $f_\theta$ will be trained using an artificially balanced dataset $\mathcal{D}'_L = \mathcal{D}_L \cup \mathcal{D}_L^s$.

Random Oversampling (ROS) is considered a classical approach to oversampling. It oversamples minority classes by randomly picking samples with replacement. It is a bootstrapping approach that, if generated in a smoothed manner (*i.e.*, by adding perturbations to the synthetic data), is also known as Random Oversampling Examples (ROSE) [54]. However, the random duplication of observations often leads to overfitting [129].

The Synthetic Minority Oversampling Technique (SMOTE) [55] attempts to address the data duplication limitation in ROS with a two stage data generation mechanism:

1. Selection phase. A minority class observation, $x^c \in C_{min}$, and one of its $k$-nearest neighbors, $x^{nn} \in C_{min}$, are randomly selected.

2. Generation phase. A synthetic observation, $x^s$, is generated along a line segment between $x^c$ and $x^{nn}$: $x^s = \alpha x^c + (1 - \alpha) x^{nn}, \alpha \sim \mathcal{U}(0, 1)$.

Although the SMOTE algorithm addresses the limitations in ROS, it brings other problems, which motivated the development of several SMOTE-based variants [57]: (1) it introduces noise when a noisy minority class observations is assigned to $x^c$ or $x^{nn}$, (2) it introduces noise when $x^c$ and $x^{nn}$ belong to different minority-class clusters, (3) it introduces near duplicate observations when $x^c$ and $x^{nn}$ are too close together and (4) it does not account for within-class imbalance (*i.e.*, different input space regions should assume a different importance according to the concentration of minority class observations).

Borderline-SMOTE [56] modifies SMOTE's selection mechanism. It calculates the $k$-nearest neighbors for all minority class observations and selects the ones that are going to be used as $x^c$ in the generation phase. An observation is selected based on the number of neighbors belonging to a different class, where the observations with no neighbors belonging to $C_{min}$ and insufficient number of neighbors belonging $C_{maj}$ are not considered for the generation phase. This approximates the synthetic observations to the border of the expected decision boundaries. Various other methods were proposed since then to modify selection mechanism, such as K-means SMOTE [68]. This approach addresses within-class imbalance and the generation of noisy synthetic data by generating data within clusters. The data generation is done according to each cluster's imbalance ratio and dispersion of minority class observations. DBSMOTE [69] also modifies the selection strategy by selecting as $x^c$ the set of core observations in a DBSCAN clustering solution.

The Adaptive Synthetic Sampling approach (ADASYN) [58] uses a comparable approach to Borderline-SMOTE. It calculates the ratio of non-minority class observations within the $k$-nearest neighbors of each $x \in C_{min}$. The amount of observations to be generated using each $x \in C_{min}$ as $x^c$ is determined according to this ratio; the more non-minority class neighbors an observation contains, the more synthetic observations are generated using it as $x^c$. The generation phase is done using the linear mechanism in SMOTE. However, this approach tends to aggravate the limitation (1) previously discussed. A second version of this method, KernelADASYN [59], replaces the generation mechanism with a weighted kernel density estimation. The weighing is done according to ADASYN's ratio and the synthetic data is sampled using the calculated Gaussian Kernel function whose bandwidth is passed as an additional hyperparameter.

Modifications to SMOTE's generation mechanism are less common and generally attempt to address problem of noisy synthetic data generation. Safe-level SMOTE [66] truncates the line segment between $x^c$ and $x^{nn}$ according to a safe level ratio. Geometric-SMOTE (G-SMOTE) [57] it generates synthetic data within a deformed and truncated hypersphere to also avoid the generation of near-duplicate synthetic data. It also introduces a modification of the selection strategy to combine the selection of majority class observations as $x^{nn}$ to avoid the introduction of noisy synthetic data.

LR-SMOTE [67] modifies both the selection and generation mechanisms. The set of observations to use as $x^c$ contains the misclassified minority class observations using a SVM classifier, out of which the potentially noisy observations are removed. The k-means clustering method is used to find the closest observations to the cluster centroids, which are used as $x^c$. The observations with a higher number of majority class neighbors are more likely to be selected as $x^{nn}$. Although the generation mechanism synthesizes observations as a linear combination between $x^c$ and $x^{nn}$, it restricts or expands this range by setting $\alpha \sim \mathcal{U}(0, M)$, where $M$ is a ratio between the average euclidean distance of each cluster's minority class observations to $x^c$ and the euclidean distance between $x^c$ and $x^{nn}$.

The Minority Oversampling Kernel Adaptive Subspaces algorithm (MOKAS) [60] adopts a different approach when compared to SMOTE-based mechanisms. It uses the adaptive subspace self-organizing map (ASSOM) [130] algorithm to learn sub-spaces (*i.e.*, different feature spaces for each unit in the SOM), out of which synthetic data is generated. The synthetic data is generated using a lower dimensional representation of the input data to ensure the reconstructed data is different from the original observations. Overall, the usage of SOMs for oversampling is uncommon. Another two examples of this approach, SOMO [61] and G-SOMO [62] use a similar approach as K-means SMOTE. In the case of G-SOMO, instead of using SMOTE's generation mechanism, it uses G-SMOTE's instead.

Oversampling via VAE [63]

Oversampling via GMM [64, 65]

Another set of network-based methods that fully replace SMOTE-based mechanisms are GAN-based architectures. One example of this approach is CGAN [70]. It uses an adversarial training approach to generate data that approximates the original data distribution and indistinguishable from the original dataset (according to the adversarial classifier). A more recent GAN-based oversampler, K-means CTGAN [71] uses a K-means clustering method as an additional attribute to train the CTGAN. In this case, cluster labels allow the reduction of within-class imbalance. These types of approaches benefit from learning the overall per-class distribution, instead of using local information only. However, GANs require more computational power to train, their performance is sensitive to the initialization and are prone to the "mode collapse" problem.

Statistical-based oversampling approaches are less common. Some methods, such as RACOG and wRACOG [74] are based on Gibbs sampling, PDFOS [76] is based on probability density function estimations and RWO [75] uses a random walk algorithm.

Although oversampling for classification problems using continuous features appears as a relatively well explored problem, there is a general lack of research on oversampling using nominal features or mixed data types (*i.e.*, using both nominal and continuous features) and regression problems. SMOTENC [55] introduces a SMOTE adaptation for mixed data types. It calculates the nearest neighbors of $x^c$ by including in the euclidean distance metric the median of the standard deviations of the continuous features for every nominal feature values that are different between $x^c$ and $x^{nn}$. The generation is done using the normal SMOTE procedure for the continuous features and the nominal features are determined with their modes within $x^c$'s nearest neighbors. The SMOTEN [55] is an oversampling algorithm for nominal features only. It uses the nearest neighbor approach proposed in Cost et al. [131] and generates $x^s$ using the modes of the features in $x^c$'s nearest neighbors. Solutions to oversampling in regression problems are generally also based on SMOTE, such as SMOTER [72] and G-SMOTER [73].

## 4.4 Active Learning

AL is an informed approach to data collection and labeling. In classification problems, when $|\mathcal{D}_U| \gg |\mathcal{D}_L|$ and it is possible to label data according to a given budget, AL methods will search for the most informative unlabeled observations. Once labeled and included into the training set, these observations are expected to improve the performance of the classifier to a greater extent when compared to randomly selecting observations. AL is an iterative process where, at each iteration, an acquisition function $f_{acq}(x, f_\theta) : \mathcal{D}_U \rightarrow \mathbb{R}$ computes a classification uncertainty score for each unlabeled observation. $f_{acq}$ provides the selection criteria based on the uncertainty scores, $f_\theta$ and the labeling budget [4].

One way to improve an AL process is via the generation of synthetic data. In this case, synthetic data is expected to improve classification with a better definition of the classifier's decision boundaries. This allows the allocation of the data collection budget over a larger area of the input space. However, research focused on this topic is both recent and limited **[CITATION]**. These methods can be divided into AL with pipelined data augmentation approaches and AL with within-acquisition data augmentation. Pipelined data augmentation is the more intuitive approach, where at each training phase data augmentation is done to improve the quality of the classifier and is independent from $f_{acq}$. In Fonseca et al. [36], the pipelined approach in tabular data achieves a superior performance compared to the traditional AL framework using the G-SMOTE algorithm and the oversampling generation policy. Other methods, although developed and tested on image data, could also be adapted for tabular data: in the Bayesian Generative Active Deep Learning framework [89] the authors propose VAEACGAN, which uses a VAE architecture along with an auxiliary-classifier generative adversarial network (ACGAN) [132] to generate synthetic data.

The Look-Ahead Data Acquisition via augmentation algorithm [4] proposes an acquisition function that considers the classification uncertainty of synthetic data generated using a given unlabeled observation, instead of only estimating classification uncertainty of the unlabeled observation itself. This approach considers both the utility of the augmented data and the utility of the unlabeled observation. This goal is achieved with the data augmentation method InfoMixup, which uses M-Mixup [78] along with the distillation of the generated synthetic data using $f_{acq}$. The authors additionally propose InfoSTN, although the original Spatial Transform Networks (STN) [133] were originally designed for image data augmentation.

## 4.5 Semi-supervised Learning

Semi-supervised learning (Semi-SL) techniques modify the learning phase of ML algorithms to leverage both labeled and unlabeled data. This approach is used when $|\mathcal{D}_U| \gg |\mathcal{D}_L|$ (similarly to AL settings), but additional labeled data is impossible or difficult to acquire. In recent years the research developed in this area directs much of its focus to neural network-based models and generative learning [38]. Overall, Semi-SL can be distinguished between transductive and inductive methods. In this section, we will focus on synthetic data generation mechanisms in inductive, perturbation-based Semi-SL algorithms applicable to tabular or feature space data.

Ladder networks [90] is semi-supervised learning architecture that learns a manifold feature space using a

Denoising Autoencoder (DAE). The synthetic data is generated during the learning phase; random noise introduced into the input data and the DAE learns to predict the original observation. Although this method was developed for image data, DAE networks can be adapted for tabular data [134].

The Π-model uses labeled and unlabeled data jointly in the training phase [91]. Besides minimizing cross-entropy, they add to the loss function the squared difference between two input level transformations (Gaussian noise and other image-specific methods) in the network's output layer (with dropout). In this case, the perturbations are applied both in the input space (via Gaussian noise) and feature space (via dropout). This model served as the source for the Mean Teacher algorithm [92], which used the same types of augmentation. The Interpolation Consistency Training (ICT) [93] method combined the mean teacher and the Mixup approach, where synthetic observations are generated using only the unlabeled observations and their predicted label using the teacher model. In Mixmatch [94], the Mixup method is used by randomly selecting any pair of observations and their true labels (if it's a labeled observation) or predicted label (if it's unlabeled).

The development of Semi-SL algorithms specifically adapted for tabular data is limited. The Semi-SL data augmentation for tabular data (SDAT) algorithm [95] uses an autoencoder to generate synthetic data in the feature space with Gaussian perturbations. The Contrastive Mixup (C-Mixup) [97] algorithm generates synthetic data using the Mixup mechanism with observation pairs within the same target label. The Mixup Contrastive Mixup algorithm (MCoM) [96] proposes the triplet Mixup method using three observations where $x^s = \lambda_i x_i + \lambda_j x_j + (1 - \lambda_i - \lambda_j) x_k$, where $\lambda_i, \lambda_j \sim \mathcal{U}(0, \alpha)$, $\alpha \in (0, 0.5]$ and $x_i$, $x_j$ and $x_k$ belong to the same target class. The same algorithm also uses the M-Mixup method as part of the feature space learning phase.

## 4.6 Self-supervised Learning

Self-supervised learning (Self-SL), although closely related to Semi-SL, assumes $\mathcal{D}_L$ to be either empty or very small. These models focus on representation learning using $\mathcal{D}_U$ using secondary learning tasks, which can be adapted to almost all types of downstream tasks [135]. This family of techniques allow the usage of raw, unlabeled data, which is generally cheaper to acquire when compared to processed, curated and labeled data. Although not all Self-SL methods rely on data augmentation (*i.e.*, STab [136]), the majority of state-of-the-art tabular Self-SL methods use data augmentation as a central concept for the training phase.

The value imputation and mask estimation method (VIME) [10] is a Semi-SL and Self-SL approach that introduces Masking, a tabular data augmentation method. It is motivated by the need to generate corrupted, difficult to distinguish synthetic data in a computationally efficient way for Self-SL training. They replace with probability $p_m$ feature values in $x_i$ with another randomly selected value of each corresponding feature. To do this, the authors use a binomial mask vector $m = [m_1, \ldots, m_d]^\perp \in \{0, 1\}^d$, $m_j \sim \text{Bern}(p_m)$, observation $x_i$ and the noise vector $\epsilon$ (*i.e.*, the vector of possible replacement values). A synthetic observation is produced as $x^s = (1 - m) \odot x_i + m \odot \epsilon$. A subsequent study proposed the SubTab [98] framework present a multi-view approach; analogous to cropping in image data or feature bagging in ensemble learning. In addition the authors propose an extension of the masking approach proposed in VIME by introducing noise using different approaches: Gaussian noise, swap-noise (*i.e.*, the approach proposed in VIME) and zero-out noise (*i.e.*, randomly replace a feature value by zero).

The Self-supervised contrastive learning using random feature corruption method (Scarf) [99] uses a similar synthetic data generation approach as VIME. Scarf differs from VIME by using contrastive loss instead of the denoising auto-encoder loss used in VIME, but this topic is out of the scope of this paper. A-SFS [100] is a Self-SL algorithm designed for feature extraction. It achieved higher performance compared to equivalent state-of-the-art augmentation-free approaches such as Tabnet [137] and uses the masking generation mechanism described in VIME.

## 5 Generation mechanisms

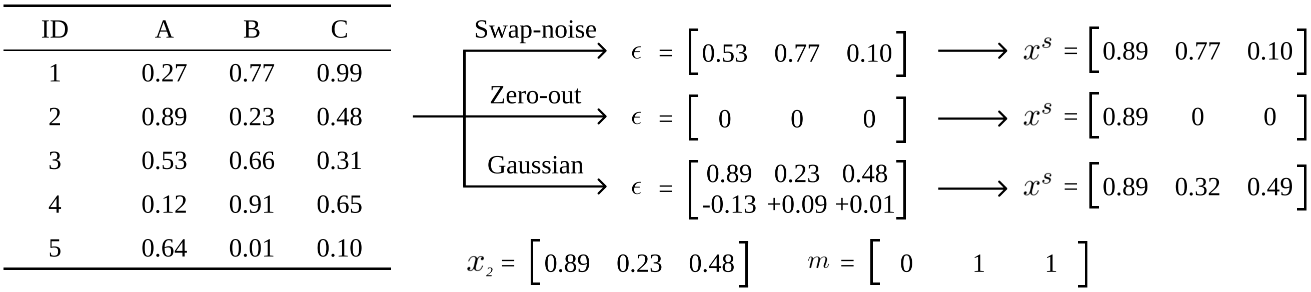
In this section we provide a general description of the synthetic data generation mechanisms found in the learning problems in Section 4. Table 3 summarizes the assumptions and usage of the generation mechanisms across the selected works and learning problems.

Table 3: Analysis of synthetic data generation mechanisms.

| Type | Mechanism | Smoothness | Manifold | Priv. | Reg. | Ovs. | AL | Semi-SL | Self-SL |
|---|---|---|---|---|---|---|---|---|---|
| Perturbation | Random | ✓ | ✓ | × | × | ✓ | × | × | × |
| | Laplace | ✓ | ✓ | ✓ | × | × | × | × | × |
| | Gaussian | ✓ | ✓ | ✓ | ✓ | × | × | ✓ | ✓ |
| | Swap-noise | × | × | × | × | × | × | ✓ | ✓ |
| | Zero-out noise | × | × | × | × | × | × | × | ✓ |
| PDF | Gaussian Gen. | × | ✓ | ✓ | × | ✓ | × | × | × |
| | Gaussian Mix. | × | ✓ | ✓ | × | ✓ | × | × | × |
| | KDE | × | ✓ | × | × | ✓ | × | × | × |
| PGM | Bayesian Net. | × | × | ✓ | ✓ | × | × | × | × |
| | Gibbs | × | × | × | ✓ | ✓ | × | × | × |
| | Random Walk | × | × | × | × | ✓ | × | × | × |
| Linear | Between-class Int. | × | ✓ | × | ✓ | × | ✓ | ✓ | × |
| | Within-class Int. | ✓ | ✓ | × | ✓ | ✓ | ✓ | ✓ | × |
| | Extrapolation | ✓ | ✓ | × | ✓ | ✓ | × | × | × |
| | Inter.+Extra. | ✓ | ✓ | × | × | ✓ | × | × | × |
| | Difference Transf. | ✓ | ✓ | × | ✓ | × | × | × | × |
| | Hard Extra. | ✓ | ✓ | × | ✓ | × | × | × | × |
| Geometric | Hypersphere | ✓ | ✓ | × | × | ✓ | ✓ | × | × |
| | Triangular | ✓ | ✓ | × | × | × | × | ✓ | × |
| | Hyperrectangle | × | ✓ | × | ✓ | × | × | × | × |
| Neural nets. | GAN | × | × | ✓ | ✓ | ✓ | ✓ | × | × |
| | AE | × | × | × | ✓ | ✓ | ✓ | ✓ | × |
| Others | Exponential M. | × | × | ✓ | × | × | × | × | × |
| | Reconstruction err. | × | × | × | × | ✓ | × | × | × |

We focus on 2 key conditions for the data generation process, smoothness and manifold space. The smoothness condition requires that if two observations $x_i, x_j$ are close, then it's expected that $y_i, y_j$ have the same value. The manifold condition requires synthetic data generation to occur within locally euclidean topological spaces. Therefore, a generation mechanism with the smoothness requirement also requires a manifold, while the opposite is not necessarily true.

In the remaining subsections we will describe the main synthetic data generation mechanisms found in the literature, based on the studies discussed in Section 4.



| ID | A | B | C |
|---|---|---|---|
| 1 | 0.27 | 0.77 | 0.99 |
| 2 | 0.89 | 0.23 | 0.48 |
| 3 | 0.53 | 0.66 | 0.31 |
| 4 | 0.12 | 0.91 | 0.65 |
| 5 | 0.64 | 0.01 | 0.10 |

Swap-noise
$\epsilon = \begin{bmatrix} 0.53 & 0.77 & 0.10 \end{bmatrix} \longrightarrow x^s = \begin{bmatrix} 0.89 & 0.77 & 0.10 \end{bmatrix}$
Zero-out
$\epsilon = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \longrightarrow x^s = \begin{bmatrix} 0.89 & 0 & 0 \end{bmatrix}$
Gaussian
$\epsilon = \begin{bmatrix} 0.89 & 0.23 & 0.48 \\ -0.13 & +0.09 & +0.01 \end{bmatrix} \longrightarrow x^s = \begin{bmatrix} 0.89 & 0.32 & 0.49 \end{bmatrix}$
$x_2 = \begin{bmatrix} 0.89 & 0.23 & 0.48 \end{bmatrix}$
$m = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix}$

Figure 2: Examples of synthetic observations generated with different masking approaches.

## 5.1 Perturbation Mechanisms

The general perturbation-based synthetic data generation mechanism is defined as $x^s = x_i + \epsilon$, where $\epsilon$ is the noise vector sampled from a certain distribution. The random perturbation mechanism can be though of as the non-informed equivalent of PGMs and PDFs. It samples $|\epsilon|$ values from a uniform distribution, *i.e.*, $e_i \sim \mathcal{U}(\cdot, \cdot), \forall e_i \in \epsilon$, while the minimum and maximum values depend on the context and level of perturbation desired, typically centered around zero.

Laplace (commonly used in DP algorithms) and Gaussian perturbations sample $\epsilon$ with $e_i \sim \text{Lap}(\cdot, \cdot)$ and $e_i \sim \mathcal{N}(\cdot, \cdot)$, respectively. Within the applications found, in the presence of categorical features, these methods tend to use n-way marginals (also known as conjunctions or contingency tables [53]) to ensure the generated data contains variability in the categorical features and the distribution of categorical feature values follows some given constraint. Although various other distributions could be used to apply perturbations, the literature found primarily focuses on introducing noise via random, Laplace and Gaussian distributions.

Masking modifies the original perturbation based approach by introducing a binomial mask vector, $m = [m_1, \ldots, m_d]^\perp \in \{0,1\}^d, m_i \sim \text{Bern}(p_m)$ and the generation mechanism is defined as $x^s = (1 - m) \odot x_i + m \odot \epsilon$ [10]. The $\epsilon$ variable is defined according to the perturbation used. The Gaussian approach generates the noise vector as $\epsilon = x_i + \epsilon'$, where $e'_i \sim \mathcal{N}(\cdot, \cdot), \forall e'_i \in \epsilon'$. The swap-noise approach shuffles the feature values from all observations to form $\epsilon$, while the zero-out noise approach sets all $\epsilon$ values to zero. Intuitively, the masking technique modifies an observation's feature values with probability $p_m$, instead of adding perturbations over the entire observation. Figure 2 shows a visual depiction of the masking technique.

## 5.2 Probability Density Function Mechanisms

The Gaussian generative model, despite unfrequently used when compared to the remaining Probaility Density Function mechanisms discussed in this subsection, is an essential building block for these mechanisms. In particular, we describe the multivariate gaussian approach, which follows near-Gaussian distribution assumptions. However, in high-dimensional data, it is possible to motivate this approach via the *Diaconis-Freedman-Meckes* effect [111], which states that high-dimensional data projections generally follow a nearly Gaussian distribution. The Gaussian generative model produces synthetic data from a Gaussian distribution $x^s \sim \mathcal{N}(\mu, \Sigma)$, where $\mu \in \mathbb{R}^d$ is a vector with the features' means and $\Sigma \in \mathbb{R}^{d \times d}$ is the covariance matrix. It follows the following density function [51]:

$$f(x) = \frac{1}{\sqrt{(2\pi)^d \text{det}(\Sigma)}} \text{exp}\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) \tag{2}$$

Consequently, to define a Gaussian generative model it is only necessary to estimate the dataset's mean and covariance matrix.

A Gaussian mixture model (GMM) comprises several Gaussians that aim to represent normally distributed subpopulations within a dataset. Its training procedure allows the model to iteratively learn the subpopulations using the Expectation Maximization algorithm. A GMM becomes more appropriate than the Gaussian generative model when the data is expected to have more than one higher-density regions, leading to a poor fit of unimodal Gaussian models.

Kernel Density Estimation (KDE) methods use a kernel function to estimate the density of the dataset's distribution at each region of the input/feature space. Despite the various kernel options, the Gaussian kernel is commonly used for synthetic data generation [59]. The general kernel estimator is defined as follows:

$$\hat{p}(x) = \frac{1}{N+h}\sum_{i=1}^{N} K\left(\frac{x-x_i}{h}\right) \tag{3}$$

Where $N = |\mathcal{D}|$, $h$ is a smoothing parameter known as bandwidth and $K$ is the kernel function. The Gaussian kernel is defined as follows:

$$G_i(x) = K\left(\frac{x-x_i}{h}\right) = \frac{1}{(\sqrt{2\pi}h)^d}\text{exp}\left(-\frac{1}{2}\frac{(x-x_i)^T(x-x_i)}{h}\right) \tag{4}$$

Therefore, the Gaussian KDE approach can also be expressed as $\hat{p}(x) = \frac{1}{N+h}\sum_{i=1}^{N} G_i(x)$, while the data is sampled from the estimated probability distribution. Figure 3 shows a visualization of the PDF mechanisms discussed, applied to a mock dataset.

## 5.3 Probabilistic Graphical Models

A Bayesian network can be thought of as a collection of as a collection of conditional distributions. It represents the joint probability distribution over the cross-product of the feature domains in $\mathcal{D}$. It is a directed acyclic graph that represents $\mathcal{D}$'s features as nodes and their conditional dependencies as directed edges. The set of features pointing directly to feature $v \in V, d = |V|$ via a single edge are known as the parent variables, $pa(v)$. A Bayesian network calculates $p(x)$ as the product of the individual density functions, based on the conditional probabilities of the parent variables:

$$p(x) = \prod_{v \in V} p(x_v|x_{pa(v)}) \tag{5}$$

Although this method requires the construction of a directed acyclic graph, there is research on ML approaches for the learning of these structures [138]. Bayesian networks can be used for synthetic data

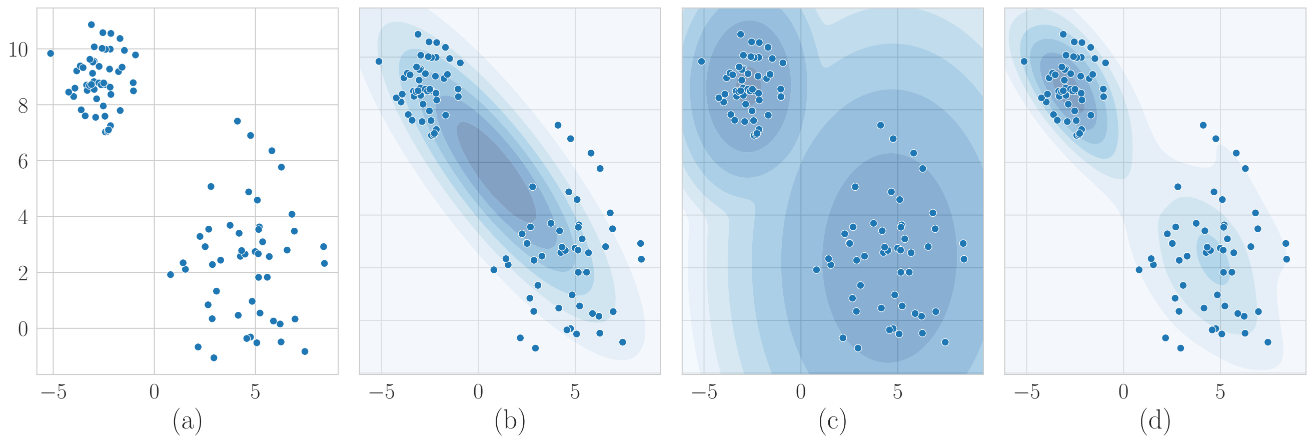


Figure 3: Examples of PDF mechanisms fitted to a mock dataset. Legend: (a) Original dataset, (b) Gaussian generative model, (c) Gaussian Mixture Model and (d) Gaussian Kernel Density Estimation.

generation when the relationship between variables is known (or can be learned) and when the data is high dimensional, making the sampling process non-trivial.

Random walk algorithms comprise the general process of iterating through a set of random steps. Although uncommon, random walk approaches may be used to sample data. The random walk approach described in Zhang et al. [75] uses the Gaussian noise mechanism over minority class observations to create synthetic observations. The Gibbs sampling mechanism also performs a random walk by iterating through sampled feature values.

Gibbs sampling is a Markov Chain Monte Carlo algorithm that iteratively samples a synthetic observation's feature values. It is a suitable method to sample synthetic data from a Bayesian network. The process starts with an initial observation selected from $\mathcal{D}$, $x_0$ and is used to begin the sampling process. In its original format, the sampling of each feature value $v$ in $x^s_i$ is conditioned by $x^s_{i-1}$ and the feature values already sampled from $x^s_i$, such that $x^s_{i,v} \sim p(x^s_{i,v}|x^s_{i,1}, \dots, x^s_{i,v-1}, x^s_{i-1,v+1}, \dots, x^s_{i-1,d})$. Therefore, Gibbs sampling is a special case of the Metropolis-Hastings algorithm.

### 5.4 Linear Transformations

Linear interpolation mechanisms can be split into two subgroups: between and within-class interpolation. Both mechanisms follow a similar approach; they use a scaling factor $\lambda$, typically sampled from either $\mathcal{U}(0, 1)$ or $\text{Beta}(\alpha, \alpha)$:

$$x^s = \lambda x_i + (1 - \lambda)x_j = x_j + \lambda(x_i - x_j) \tag{6}$$

The within-class interpolation mechanism selects two observations from the same class, while the between-class interpolation mechanism selects two observations from different classes and also interpolates the one-hot encoded target classes $y_i$ and $y_j$. However, the approach to select observations might vary according to the ML task and data generation algorithm. For example, most SMOTE-based methods

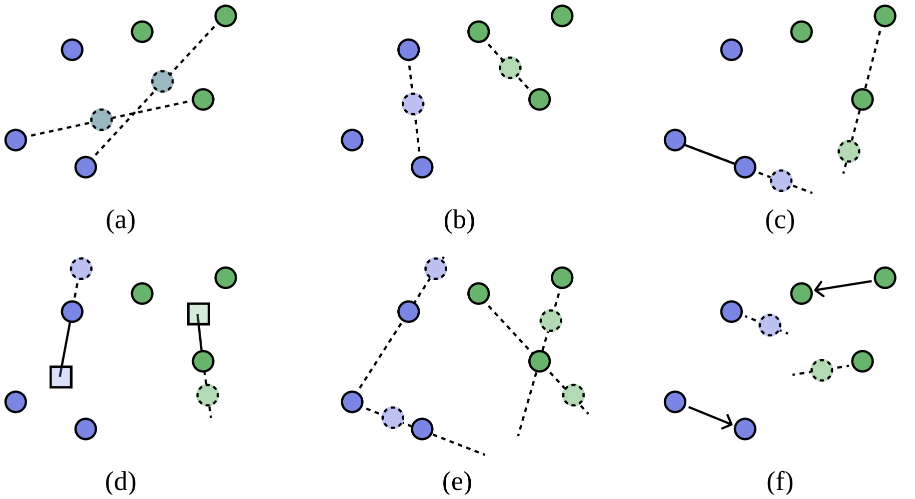


Figure 4: Examples of linear transformation mechanisms. Legend: (a) Between-class interpolation, (b) Within-class interpolation, (c) Observation-based extrapolation, (d) Mean-based extrapolation, (e) Combination of interpolation and extrapolation and (f) Difference transform.

select a center observation and a random observation within its $k$-nearest neighbors belonging to the same class, while the Mixup method selects two random observations, regardless of their class membership.

The observation-based linear extrapolation mechanism modifies Equation 6 such that $x^s = x_i + \lambda(x_i - x_j)$, while the mean-based extrapolation mechanism uses the mean of a class' observations, $\mu^c$ and a randomly selected observation to generate $x^s = x_i^c + \lambda(x_i^c - \mu^c)$. The combination of both interpolation and extrapolation mechanisms was found in the literature. This mechanism can be achieved using Equation 6 and modifying $\lambda$'s range to either decreasing its minimum value below zero or increasing its maximum value above one.

The difference transform mechanism uses two observations to compute a translation vector (multiplied by the scaling factor $\lambda$) and apply it on a third observation:

$$x^s = x_i + \lambda(x_j - x_k) \tag{7}$$

Although there are various linear transformation mechanisms in the literature, the majority of the relevant studies found applied linear interpolation mechanisms. Within-class interpolation was frequently found in oversampling methods, while between-class interpolation was found most often in regularization methods. A depiction of the linear transformation mechanisms found in the literature are presented in Figure 4.

## 5.5 Geometric Transformations

Overall, geometric transformation mechanisms were not frequently found in the literature. They are primarily used to develop Mixup or SMOTE-based variants. Figure 5 shows a visual example of the related mechanisms.

The hypersphere mechanism generates data within a distorted, n-dimensional hyperspheroid. It is formed using an observation to define the center of the geometry and another to define its edge. It is defined with two hyperparameters, the deformation factor, $\alpha_{def} \in [0, 1]$, and the truncation factor, $\alpha_{trunc} \in [-1, 1]$. The

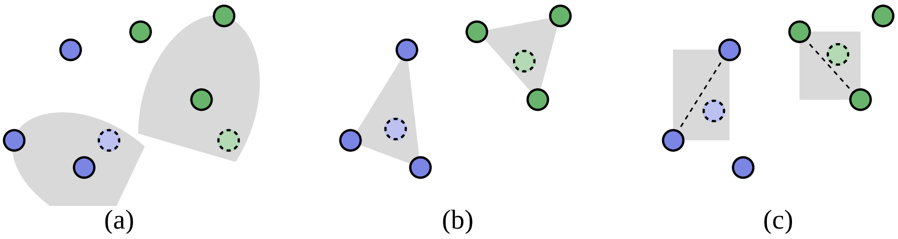


Figure 5: Examples of geometric transformation mechanisms. Legend: (a) hypersphere mechanism, (b) triangular mechanism and (c) hyperrectangle mechanism.

deformation factor deforms the hypersphere into an elliptic shape, where $\alpha_{def} = 1$ applies no deformation and $\alpha_{def} = 0$ creates a line segment. The truncation factor limits the generation area of the hyperspheroid within a subset of the hypersphere, where $\alpha_{trunc} = 0$ applies no truncation, $\alpha_{trunc} = 1$ uses the half of the area between the two selected observations and $\alpha_{trunc} = -1$ uses the opposing area. In Figure 5a, the two generation areas were formed using approximately $\alpha_{trunc} = \alpha_{def} = 0.5$.

The triangular mechanism selects three observations to generate $x^s = \lambda_i x_i + \lambda_j x_j + (1 - \lambda_i - \lambda_j)x_k$, where $\lambda_i, \lambda_j \sim \mathcal{U}(0, \alpha)$, $\alpha \in (0, 0.5]$. The hyperrectangle mechanism uses an approach similar similar to Equation 6. However, the scaling factor is changed into a scaling vector, $\Lambda = [\lambda_1, \ldots, \lambda_d] \in [0, 1]^d, \lambda_i \sim \text{Beta}(\alpha, \alpha)$, where $\alpha$ is a hyperparameter used to define the Beta distribution. A synthetic observation is generated with $x^s = \Lambda \odot x_i + (1 - \Lambda) \odot x_j$, where $\odot$ denotes the Hadamard product. This operation originates a generation area like the ones presented in Figure 5c.

## 5.6 Neural Networks

Generative Adversarial Network (GAN) architectures are structured as a minimax two-player game composed of two models, a generator and a discriminator. Both models are trained simultaneously throughout the learning phase to learn to generate data with similar statistical properties when compared to the original data. The generative model captures the data distribution, while the discriminator estimates the probability of an observation coming from the training data. The goal of the generator model is to produce synthetic observations that are capable of fooling the discriminator, making it impossible for the discriminator to distinguish real from synthetic observations. Although they were originally developed in an unsupervised learning setting [139], subsequent contributions proposed GANs for semi-SL, supervised learning and reinforcement learning.

An autoencoder (AE) is a type of neural network architecture that learns manifold representations of an input space. These models are typically trained by regenerating the input and are designed with a bottleneck in the hidden layers that corresponds to the learned feature space. It contains two parts, an encoder and a decoder. The encoder transforms the input data into lower-dimensional representations (*i.e.*, the feature space), while the decoder projects these representations into the original input space. Since it was first proposed [140], many variants were developed for multiple purposes. However, based on the literature found for synthetic data generation, the variational AE architecture appears to be the most popular approach.

# 6 Evaluating the Quality of Synthetic Data

The vast majority of synthetic data generation models are evaluated on a ML utility basis. There is a general lack of research on the development of metrics to evaluate the quality of synthetic data beyond common metrics such as Overall Accuracy (OA) or F1-score. One motivation to do this is the ability to anticipate the quality of the data after training a ML classifier, which may be an expensive and time-consuming task. This is a challenging problem, since the usefulness of synthetic data generators depend on the assumptions imposed according to the dataset, domain and ML problem [141]. This section focuses on the main evaluation approaches found in the literature, as well as recently proposed methods. For a more comprehensive analysis of performance metrics for synthetic data evaluation, the reader is referred to [142] and [12].

The GANBLR model [85] was evaluated on three aspects: (1) ML utility, (2) Statistical similarity, and (3) Interpretability. In Xu et al. [87], the authors evaluate the CTGAN and TVAE models using a likelihood fitness metric (to measure statistical similarity) and ML efficacy (*i.e.*, utility). Hittmeir et al. [143] evaluate synthetic data generators using a 2-step approach: Similarity comparison and data utility. According to Alaa et al. [13], the evaluation of generative models should quantify three key aspects of synthetic data:

1. Fidelity TODO: DEFINE

2. Diversity

3. Generalization

The effective evaluation of synthetic data generation methods is a complex task. A good performance with respect to one evaluation method does not necessarily imply a good performance on the primary ML task, results from different evaluation methods seem to be independent and evaluating the models directly onto the target application is generally recommended [12]. Each evaluation procedure must be carefully implemented and adapted according to the use case.

## 6.1 Quantitative approaches

The Kullback-Leibler (KL) divergence (and equivalently the log-likelihood) is a common approach to evaluate generative models [12]. Other commonly used metrics, like Parzen window estimates, appear to be a generally poor quality estimation method and are not recommended for most applications Theis et al. [12]. KL divergence is defined as follows:

$$D_{KL}(P||Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)} \tag{8}$$

Where $\mathcal{X}$ is a probability space, $P$ and $Q$ are probability distributions based on $\mathcal{D}$ and $\mathcal{D}^s$, respectively. KL divergence is a non symmetric measurement that represents how a reference/true probability distribution ($P$) differs from another ($Q$). A $D_{KL}$ close to zero means $Q$ is similar to $P$. However, metrics like the KL divergence or the log-likelihood are generally difficult to interpret, does not scale well for high dimensional data and fails to highlight model failures [13]. Another related metric, used in [144], is the Jensen-Shannon (JS) divergence. It consists of a symmetrized and smoothed variation adaptation of the KL divergence. Having $M = \frac{P+Q}{2}$, it is calculated as:

$$D_{JS}(P||Q) = \frac{D_{KL}(P||M) + D_{KL}(Q||M)}{2} \tag{9}$$

The Wasserstein Distance is another commonly used metric to estimate the distance between two distribution functions. It was also used to develop GAN variants since it improves the stability in the training of GANs [145, 146].

The propensity score was considered an appropriate performance metric to measure the utility of masked data [147]. This metric is estimated via the training of a classifier (typically a logistic regression) on a dataset merged with both the original and synthetic data and labeled information regarding the source of each observation (synthetic or original), to predict the likelihood of an observation to be synthetic. Therefore, this approach guarantees observation-level insights regarding the faithfulness of each observation. Woo et al. [147] suggest a summarization of this metric by computing:

$$U_p = \frac{1}{N}\sum_{i=1}^{N}(\hat{p}_i - c)^2 \tag{10}$$

Where $N = |\mathcal{D} \cup \mathcal{D}^s|$, $c = \frac{|\mathcal{D}^s|}{N}$ and $\hat{p}_i$ is the estimated propensity score for unit $i$. This metric is also defined as the propensity Mean Squared Error (pMSE) [141]. When a synthetic dataset is relatively similar to the original dataset, $U_p$ will be closer to zero. Furthermore, if the synthetic data is indistinguishable from the real data, the expected pMSE value is given by [148]:

$$E(pMSE) = \frac{(k-1)(1-c)^2 c}{N} \tag{11}$$

Where $k$ is the number of parameters in the logistic regression model (including bias). When the synthetic dataset is easily distinguishable from the original dataset, $U_p$ will be close to $(1-c)^2$. Dankar et al. [30], established a generally consistent, weak negative correlation between $U_p$ and OA.

Chundawat et al. [141] proposed TabSynDex to address the lack of uniformity of synthetic data evaluation, which can also be used as a loss function to train network-based models. It is a single metric evaluation approach bounded within $[0, 1]$ that consists of a combination between (1) the relative errors of basic statistics (mean, median and standard deviation), (2) the relative errors of correlation matrices, (3) a pMSE-based index, (4) a support coverage-based metric for histogram comparison and (5) the performance difference in a ML efficacy-based metric between models trained on real and synthetic data.

The three-dimensional metric proposed by Alaa et al. [13] presents an alternative evaluation approach. It combines three metrics ($\alpha$-Precision, $\beta$-Recall and Authenticity) for various application domains. It extends the Precision and Recall metrics defined in [149] into $\alpha$-Precision and $\beta$-Recall, which are used to quantify fidelity and diversity. Finally, the authenticity metric is estimated using a classifier that is trained based on the distance (denoted as $d$) between $x^s$ and its nearest neighbor in $\mathcal{D}$, $x_{i^*}$; if $d(x^s, x_{i^*})$ is smaller than the distance between $x_{i^*}$ and its nearest neighbor in $\mathcal{D}\backslash\{x_{i^*}\}$, $x^s$ will likely be considered unauthentic. This approach provides a three fold perspective over the quality of $\mathcal{D}^s$ and allows a sample-level analysis of the generator's performance. Furthermore, there is a relative trade-off between the two metrics used to audit the generator and the synthetic data; a higher $\alpha$-Precision score will generally correspond to a lower Authenticity score and vice versa.

A less common evaluation approach is to attempt to replicate the results of studies using synthetic data [150, 151, 152]. Another method is the computation of the average distance among synthetic observations and their nearest neighbors within the original dataset [143]. The Confidence Interval Overlap and Average Percentage Overlap metrics may be used to evaluate synthetic data specifically for regression problems [153, 154].

## 6.2 Qualitative approaches

One of the approaches found in the literature is the comparison of the features' distributions with synthetic data and the original data using histogram plots [143]. This comparison can be complemented with the quantification of these distribution differences [150]. Another evaluation method is the comparison of correlation matrices via heat map plots [143].

Another way to assess the quality of synthetic data is the subjective assessment by domain experts [150]. The goal of such test is to understand whether domain experts are able to distinguish synthetic from real data, which could be quantified with classification performance metrics. A low classification performance implies synthetic data that is difficult to distinguish from real data.

# 7 Discussion

The generation of tabular and feature space synthetic data has applications in multiple ML tasks and domains. Specifically, we found six areas that were proven to benefit from synthetic data: data privacy, regularization, oversampling, active learning, semi-supervised learning and self-supervised learning. Synthetic data may be used either as an accessory task to improve a ML model's performance over a primary task (*e.g.*, regularization and oversampling), an intermediate task (*e.g.*, feature extraction), or as a final product itself (*e.g.*, data anonymization). The analysis of data generation algorithms for each relevant learning problem led to the proposal of a general purpose taxonomy primarily focused on the underlying mechanisms used for data generation. We characterized every algorithm discussed in this work into four categories: (1) architecture, (2) application level, (3) data space and (4) scope.

This work focused primarily on the mechanisms used to generate synthetic observations; preprocessing, learning phase design, feature space learning and ML task-specific contributions were secondary objectives for analysis. Consequently, understanding of how the constraints within each task condition the choice and design of the synthetic data generator is a subject of future work.

The successful implementation of synthetic data generation generally requires a few considerations:

1. Ensuring the dataset's features are comprised within similar, fixed boundaries. For example, any method using a neighbors-based approach will rely on distance measurements (typically the euclidean distance), which is sensitive to the scale of the data and a nearest-neighbors estimation may vary depending on whether the data was scaled *a priori*. This can be achieved with data scaling.

2. Various generation mechanisms require a manifold. There are two approaches to address non-manifold input data: (1) Adopt methods sensitive to the presence of non-metric features, or (2) project the input data into a manifold (*i.e.*, a feature space).

3. The smoothness assumption is prevalent in linear and perturbation-based data generation mechanisms. If a classification problem has low class separation and difficult to solve, the choice in the design of the generator algorithm is also difficult. Generally, generation algorithm with a global scope might adapt better to classification problems with low separability. On the other hand, problems with higher separability might require a definition of more uniform decision boundaries to prevent overfitting, which can be achieved with generation algorithms with a local scope.

4. Considering the trade-off between performance and computational power. It is generally understood that computationally-intensive approaches tend to produce synthetic data with higher quality. When trained properly, neural network mechanisms typically lead to synthetic data that is more difficult to distinguish compared to the remaining approaches. Geometric mechanisms have also achieved good results but often require a careful tuning of their hyperparameters. Linear and perturbation mechanisms do not require much training and use less hyperparameters, but have been know for often producing low diversity synthetic data (*vis a vis* the original dataset).

Throughout the analysis of the literature, we identified six types of generation mechanisms and discuss the mechanisms used in classical and state-of-the-art techniques.

# 8 Future Work

Research across the different applications appears to be sandboxed even though all techniques integrate synthetic data in its core.

It is generally understood that, if learned properly, the feature space is expected to be convex and isotropic. In that case, using linear generation techniques in the feature space would produce synthetic data without introducing noise [81]. However, it is unclear which types of model/architectures and training procedures contribute to the learning of a good feature space according to the context.

Given the breadth and complexity of input-level and feature-level data generation mechanisms, it is increasingly important to find a method to efficiently determine the most appropriate data generation policies. However, the complexity of this task is determined by various factors: different data types, ML problems, model architectures, computational resources, performance metrics and contextual constraints. Auto-augmentation and meta learning aim to address this challenge and are still subject to active research.

The quality of synthetic data generation in high-dimensional domains appears as a prevailing limitation in most applications. This method might be addressed with dimensionality reduction techniques along with data generation in the feature space. However, research on generation in the feature space is greatly focused on GAN architectures, which require significant computational power. Other methods for learning manifold space embeddings could be explored to address this limitation.

To the best of our knowledge, research on data augmentation using auto-encoder architectures is sparse. There is, however, a few studies performing data augmentation in different domains using tabular data [88]. More commonly, autoencoders are used to learn a manifold features space for more complex data types. As long as the method used to generate the feature space is appropriate, the methods discussed in this study can be used in the feature space regardless of the type of data.

It remains an open question which feature space transformations, or types of transformations, create better synthetic data [81].

There is not much research concerning the quality and general performance between data generation on the input, feature and output space.

The evaluation of anonymization techniques lack standardized, objective and reliable performance metrics and benchmark datasets to allow an easier comparison across classifiers to evaluate key aspects of data anonymization (resemblance, utility, privacy and performance). These datasets should contain mixed data types (*i.e.*, a combination of categorical, ordinal, continuous and discrete features) and the metrics should evaluate the performance of different data mining tasks along with the anonymization reliability. This problem appears to be universal across domains. For example, Hernandez et al. [15] observed the lack of a universal method or metric to report the performance synthetic data generation algorithms for tabular health records. Therefore, in order to facilitate the usage of these techniques in industry domains, these benchmarks must also be realistic. Rosenblatt et al. [44] attempts to address this problem by proposing a standardized evaluation methodology using standard datasets and real-world industry applications.

Computational cost and inconsistent quality of synthetic data generated with GANs (*e.g.*, mode collapse).

Interpretability of representations when generating data in the feature space. The ability to interpret feature space representations could guide the design of data generation techniques.

Research on differentially private variational autoencoders is sparse to non-existent. The only related study found in the literature was developed in [155]. However, it is not peer reviewed or particularly popular, which led us to discard this paper from our analysis.

Unlike with data privacy solutions, data augmentation techniques generally do not consider the similarity/dissimilarity of synthetic data. The study of quality metrics for supervised learning may reduce computational overhead and experimentation time. No studies related to the relationship of quality metrics and performance in the primary ML task were found **[CONFIRM!!!]**.

There is not a clear understanding of what types of data augmentation methods are more appropriate according to different model architectures, ML tasks or domains and the reason why they work better or worse depending on the task. In addition, it is still unclear *why* data augmentation works. Research on this topic lacks depth and fails to address the theoretical underpinnings [7].

In some domains, a common approach for data augmentation is the combination of several data augmentation methods to increase the diversification of synthetic data. This is true for both text classification [18] and image classification **[CITATION]**. However, for tabular data, no similar approach was found.

"Dao et al. (2019) note that "data augmentation is typically performed in an ad-hoc manner with little understanding of the underlying theoretical principles", and claim the typical explanation of DA as regularization to be insufficient." [7]

There is a lack of research on oversampling solutions to generate synthetic data with mixed data types and datasets with exclusively non metric features.

There is a lack of methods adapted to use categorical features for tabular data.

There is a lack of methods directed to regression problems.

There is a paucity of research on the usage of probabilistic-based generation mechanisms in oversampling.

To the best of our knowledge, research on few-shot learning for tabular data is residual to non-existent. Few-shot learning research using synthetic data generation techniques has been extensively developed using image [156, 157] and text data [158], but they are rarely adapted or tested in tabular data.

There is no clear understanding of the most appropriate data augmentation techniques used to train self-supervised models and how their behavior and performance varies according to the data generation method used.

Oversampling does not seem to be a relevant source of bias in behavioral research and does not appear to have an appreciably different effect on results for directly versus indirectly oversampled variables [159]. However, most oversampling methods do not account for the distribution in $\mathcal{D}$, which is especially important for features with sensitive information (*e.g.*, gender or ethnicity). Therefore, the application of oversampling methods on user data may further increase the bias in classification/discrimination between gender or ethnicity groups.

The combination of data generation strategies is an approach commonly found in different problems, such as self-supervised learning [5]. It can be more frequently found in text data applications [18] and image data **[CITATION]**. Although common in synthetic data generation applications for image data, there is a lack of studies on the potential of ensembles of generation mechanisms on tabular data, *i.e.*, understanding how selecting with different probabilities different generation mechanisms to generate synthetic data would affect the performance of the primary ML task.

## 8.1 Research directions

Quantifying the quality of the generated data:

1. Realistic
2. Similarity
3. Usefulness (determine purpose and relevant performance metric)
4. Understand the relationship between the 3 factors

# 9 Conclusions

# References

[1] Samuel A Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E Tillman, Prashant Reddy, and Manuela Veloso. “Generating synthetic data in finance: opportunities, challenges and pitfalls”. In: *Proceedings of the First ACM International Conference on AI in Finance*. 2020, pp. 1–8.

[2] Samuli Laine and Timo Aila. "Temporal ensembling for semi-supervised learning". In: *International Conference on Learning Representations (ICLR)*. Vol. 4. 5. 2017, p. 6.

[3] Joao Fonseca, Georgios Douzas, and Fernando Bacao. "Improving imbalanced land cover classification with K-Means SMOTE: Detecting and oversampling distinctive minority spectral signatures". In: *Information* 12.7 (2021), p. 266.

[4] Yoon-Yeong Kim, Kyungwoo Song, JoonHo Jang, and Il-Chul Moon. "LADA: Look-Ahead Data Acquisition via Augmentation for Deep Active Learning". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 22919–22930.

[5] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. "Bootstrap your own latent-a new approach to self-supervised learning". In: *Advances in neural information processing systems* 33 (2020), pp. 21271–21284.

[6] Jiang-Jing Lv, Xiao-Hu Shao, Jia-Shui Huang, Xiang-Dong Zhou, and Xi Zhou. "Data augmentation for face recognition". In: *Neurocomputing* 230 (2017), pp. 184–196.

[7] Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. "A Survey of Data Augmentation Approaches for NLP". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Online: Association for Computational Linguistics, Aug. 2021, pp. 968–988.

[8] Talha Mahboob Alam, Kamran Shaukat, Ibrahim A Hameed, Suhuai Luo, Muhammad Umer Sarwar, Shakir Shabbir, Jiaming Li, and Matloob Khushi. "An investigation of credit card default prediction in the imbalanced datasets". In: *IEEE Access* 8 (2020), pp. 201173–201198.

[9] Terrance DeVries and Graham W Taylor. "Dataset augmentation in feature space". In: *arXiv preprint arXiv:1702.05538* (2017).

[10] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. "Vime: Extending the success of self-and semi-supervised learning to tabular domain". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 11033–11043.

[11] Diederik P Kingma, Max Welling, et al. "An introduction to variational autoencoders". In: *Foundations and Trends® in Machine Learning* 12.4 (2019), pp. 307–392.

[12] L Theis, A van den Oord, and M Bethge. "A note on the evaluation of generative models". In: *International Conference on Learning Representations (ICLR 2016)*. 2016, pp. 1–10.

[13] Ahmed Alaa, Boris Van Breugel, Evgeny S Saveliev, and Mihaela van der Schaar. "How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 290–306.

[14] Miro Mannino and Azza Abouzied. "Is this real? Generating synthetic data that looks real". In: *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 2019, pp. 549–561.

[15] Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. "Synthetic Data Generation for Tabular Health Records: A Systematic Review". In: *Neurocomputing* (2022).

[16] Trivellore E Raghunathan. "Synthetic data". In: *Annual Review of Statistics and Its Application* 8 (2021), pp. 129–140.

[17] Jakub Nalepa, Michal Marcinkiewicz, and Michal Kawulok. "Data augmentation for brain-tumor segmentation: a review". In: *Frontiers in computational neuroscience* 13 (2019), p. 83.

[18] Markus Bayer, Marc-André Kaufhold, and Christian Reuter. "A survey on data augmentation for text classification". In: *ACM Computing Surveys* (2021).

[19] Connor Shorten, Taghi M Khoshgoftaar, and Borko Furht. "Text data augmentation for deep learning". In: *Journal of big Data* 8.1 (2021), pp. 1–34.

[20] Jiaao Chen, Derek Tam, Colin Raffel, Mohit Bansal, and Diyi Yang. “An empirical survey of data augmentation for limited data learning in NLP”. In: *arXiv preprint arXiv:2106.07499* (2021).

[21] Pei Liu, Xuemin Wang, Chao Xiang, and Weiye Meng. “A survey of text data augmentation”. In: *2020 International Conference on Computer Communication and Network Security (CCNS)*. IEEE. 2020, pp. 191–195.

[22] Xin Yi, Ekta Walia, and Paul Babyn. “Generative adversarial network in medical imaging: A review”. In: *Medical image analysis* 58 (2019), p. 101552.

[23] Xiang Wang, Kai Wang, and Shiguo Lian. “A survey on face data augmentation for the training of deep neural networks”. In: *Neural computing and applications* 32.19 (2020), pp. 15503–15531.

[24] Connor Shorten and Taghi M Khoshgoftaar. “A survey on image data augmentation for deep learning”. In: *Journal of big data* 6.1 (2019), pp. 1–48.

[25] Cherry Khosla and Baljit Singh Saini. “Enhancing performance of deep learning models with different data augmentation techniques: A survey”. In: *2020 International Conference on Intelligent Engineering and Management (ICIEM)*. IEEE. 2020, pp. 79–85.

[26] Brian Kenji Iwana and Seiichi Uchida. “An empirical survey of data augmentation for time series classification with neural networks”. In: *Plos one* 16.7 (2021), e0254841.

[27] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. “Time series data augmentation for deep learning: a survey”. In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. Ed. by Zhi-Hua Zhou. International Joint Conferences on Artificial Intelligence Organization, Aug. 2021, pp. 4653–4660.

[28] Tong Zhao, Gang Liu, Stephan Günnemann, and Meng Jiang. “Graph Data Augmentation for Graph Machine Learning: A Survey”. In: *arXiv preprint arXiv:2202.08871* (2022).

[29] Nour Eldeen Khalifa, Mohamed Loey, and Seyedali Mirjalili. “A comprehensive survey of recent trends in deep learning for digital images augmentation”. In: *Artificial Intelligence Review* (2021), pp. 1–27.

[30] Fida K Dankar and Mahmoud Ibrahim. “Fake it till you make it: Guidelines for effective synthetic data generation”. In: *Applied Sciences* 11.5 (2021), p. 2158.

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

[32] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. “Understanding deep learning (still) requires rethinking generalization”. In: *Communications of the ACM* 64.3 (2021), pp. 107–115.

[33] Yi Zeng, Han Qiu, Gerard Memmi, and Meikang Qiu. “A data augmentation-based defense method against adversarial attacks in neural networks”. In: *International Conference on Algorithms and Architectures for Parallel Processing*. Springer. 2020, pp. 274–289.

[34] John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. “Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp”. In: *arXiv preprint arXiv:2005.05909* (2020).

[35] José A Sáez, Bartosz Krawczyk, and Michał Woźniak. “Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets”. In: *Pattern Recognition* 57 (2016), pp. 164–178.

[36] Joao Fonseca, Georgios Douzas, and Fernando Bacao. “Increasing the Effectiveness of Active Learning: Introducing Artificial Data Generation in Active Learning for Land Use/Land Cover Classification”. In: *Remote Sensing* 13.13 (2021), p. 2619.

[37] Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and Wlliam Campbell. “A Closer Look At Feature Space Data Augmentation For Few-Shot Intent Classification”. In: *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*. 2019, pp. 1–10.

[38] Jesper E Van Engelen and Holger H Hoos. “A survey on semi-supervised learning”. In: *Machine Learning* 109.2 (2020), pp. 373–440.

[39] Ryan McKenna, Gerome Miklau, and Daniel Sheldon. “Winning the NIST Contest: A scalable and general approach to differentially private synthetic data”. In: *Journal of Privacy and Confidentiality* 11.3 (2021).

[40] Moritz Hardt, Katrina Ligett, and Frank McSherry. “A simple and practical algorithm for differentially private data release”. In: *Proceedings of the 25th International Conference on Neural Information Processing Systems-Volume 2*. 2012, pp. 2339–2347.

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

[42] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. “Privbayes: Private data release via bayesian networks”. In: *ACM Transactions on Database Systems (TODS)* 42.4 (2017), pp. 1–41.

[43] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. “Differentially private generative adversarial network”. In: *arXiv preprint arXiv:1802.06739* (2018).

[44] Lucas Rosenblatt, Xiaoyan Liu, Samira Pouyanfar, Eduardo de Leon, Anuj Desai, and Joshua Allen. “Differentially private synthetic data: Applied evaluations and enhancements”. In: *arXiv preprint arXiv:2011.05537* (2020).

[45] James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. “PATE-GAN: Generating synthetic data with differential privacy guarantees”. In: *International conference on learning representations*. 2018.

[46] Giuseppe Vietri, Grace Tian, Mark Bun, Thomas Steinke, and Steven Wu. “New oracle-efficient algorithms for private synthetic data release”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 9765–9774.

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

[48] Christopher De Sa, Ihab Ilyas, Benny Kimelfeld, Christopher Re, and Theodoros Rekatsinas. “A Formal Framework for Probabilistic Unclean Databases”. In: *22nd International Conference on Database Theory (ICDT 2019)*. 2019.

[49] Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. “Probabilistic databases”. In: *Synthesis lectures on data management* 3.2 (2011), pp. 1–180.

[50] Chang Ge, Shubhankar Mohapatra, Xi He, and Ihab F Ilyas. “Kamino: constraint-aware differentially private data synthesis”. In: *Proceedings of the VLDB Endowment* 14.10 (2021), pp. 1886–1899.

[51] Thee Chanyaswad, Changchang Liu, and Prateek Mittal. “Ron-gauss: Enhancing utility in non-interactive private data release”. In: *Proceedings on Privacy Enhancing Technologies* 2019.1 (2019), pp. 26–46.

[52] Ryan McKenna, Gerome Miklau, Michael Hay, and Ashwin Machanavajjhala. “Optimizing error of high-dimensional statistical queries under differential privacy”. In: *Proceedings of the VLDB Endowment* 11.10 (2018).

[53] Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Zhiwei Steven Wu. "Dual query: Practical private query release for high dimensional data". In: *International Conference on Machine Learning*. PMLR. 2014, pp. 1170–1178.

[54] Giovanna Menardi and Nicola Torelli. "Training and assessing classification rules with imbalanced data". In: *Data mining and knowledge discovery* 28.1 (2014), pp. 92–122.

[55] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[56] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning". In: *International conference on intelligent computing*. Springer. 2005, pp. 878–887.

[57] Georgios Douzas and Fernando Bacao. "Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE". In: *Information Sciences* 501 (2019), pp. 118–135.

[58] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. "ADASYN: Adaptive synthetic sampling approach for imbalanced learning". In: *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. IEEE. 2008, pp. 1322–1328.

[59] Bo Tang and Haibo He. "KernelADASYN: Kernel based adaptive synthetic data generation for imbalanced learning". In: *2015 IEEE congress on evolutionary computation (CEC)*. IEEE. 2015, pp. 664–671.

[60] Chin-Teng Lin, Tsung-Yu Hsieh, Yu-Ting Liu, Yang-Yin Lin, Chieh-Ning Fang, Yu-Kai Wang, Gary Yen, Nikhil R Pal, and Chun-Hsiang Chuang. "Minority oversampling in kernel adaptive subspaces for class imbalanced datasets". In: *IEEE Transactions on Knowledge and Data Engineering* 30.5 (2017), pp. 950–962.

[61] Georgios Douzas and Fernando Bacao. "Self-Organizing Map Oversampling (SOMO) for imbalanced data set learning". In: *Expert systems with Applications* 82 (2017), pp. 40–52.

[62] Georgios Douzas, Rene Rauch, and Fernando Bacao. "G-SOMO: An oversampling approach based on self-organized maps and geometric SMOTE". In: *Expert Systems with Applications* 183 (2021), p. 115230.

[63] Wangzhi Dai, Kenney Ng, Kristen Severson, Wei Huang, Fred Anderson, and Collin Stultz. "Generative oversampling with a contrastive variational autoencoder". In: *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2019, pp. 101–109.

[64] Meng Xing, Yanbo Zhang, Hongmei Yu, Zhenhuan Yang, Xueling Li, Qiong Li, Yanlin Zhao, Zhiqiang Zhao, and Yanhong Luo. "Predict DLBCL patients' recurrence within two years with Gaussian mixture model cluster oversampling and multi-kernel learning". In: *Computer Methods and Programs in Biomedicine* 226 (2022), p. 107103.

[65] Zhaozhao Xu, Derong Shen, Yue Kou, and Tiezheng Nie. "A Synthetic Minority Oversampling Technique Based on Gaussian Mixture Model Filtering for Imbalanced Data Classification". In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[66] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem". In: *Pacific-Asia conference on knowledge discovery and data mining*. Springer. 2009, pp. 475–482.

[67] XW Liang, AP Jiang, T Li, YY Xue, and GT Wang. "LR-SMOTE—An improved unbalanced data set oversampling based on K-means and SVM". In: *Knowledge-Based Systems* 196 (2020), p. 105845.

[68] Georgios Douzas, Fernando Bacao, and Felix Last. "Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE". In: *Information Sciences* 465 (2018), pp. 1–20.

[69] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. "DBSMOTE: density-based synthetic minority over-sampling technique". In: *Applied Intelligence* 36.3 (2012), pp. 664–684.

[70] Georgios Douzas and Fernando Bacao. "Effective data generation for imbalanced learning using conditional generative adversarial networks". In: *Expert Systems with applications* 91 (2018), pp. 464–471.

[71] Chunsheng An, Jingtong Sun, Yifeng Wang, and Qingjie Wei. "A K-means Improved CTGAN Oversampling Method for Data Imbalance Problem". In: *2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*. IEEE. 2021, pp. 883–887.

[72] Luís Torgo, Rita P Ribeiro, Bernhard Pfahringer, and Paula Branco. "Smote for regression". In: *Portuguese conference on artificial intelligence*. Springer. 2013, pp. 378–389.

[73] Luís Camacho, Georgios Douzas, and Fernando Bacao. "Geometric SMOTE for regression". In: *Expert Systems with Applications* (2022), p. 116387.

[74] Barnan Das, Narayanan C Krishnan, and Diane J Cook. "RACOG and wRACOG: Two probabilistic oversampling techniques". In: *IEEE transactions on knowledge and data engineering* 27.1 (2014), pp. 222–234.

[75] Huaxiang Zhang and Mingfang Li. "RWO-Sampling: A random walk over-sampling approach to imbalanced data classification". In: *Information Fusion* 20 (2014), pp. 99–116.

[76] Ming Gao, Xia Hong, Sheng Chen, Chris J Harris, and Emad Khalaf. "PDFOS: PDF estimation based over-sampling for imbalanced two-class problems". In: *Neurocomputing* 138 (2014), pp. 248–259.

[77] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. "mixup: Beyond Empirical Risk Minimization". In: *International Conference on Learning Representations*. 2018.

[78] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. "Manifold mixup: Better representations by interpolating hidden states". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6438–6447.

[79] Hongyu Guo. "Nonlinear mixup: Out-of-manifold data augmentation for text classification". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 4044–4051.

[80] Xiexing Feng, QM Jonathan Wu, Yimin Yang, and Libo Cao. "An autuencoder-based data augmentation strategy for generalization improvement of DCNNs". In: *Neurocomputing* 402 (2020), pp. 283–297.

[81] Tsz-Him Cheung and Dit-Yan Yeung. "Modals: Modality-agnostic automated data augmentation in the latent space". In: *International Conference on Learning Representations*. 2020.

[82] Xiaofeng Liu, Yang Zou, Lingsheng Kong, Zhihui Diao, Junliang Yan, Jun Wang, Site Li, Ping Jia, and Jane You. "Data augmentation via latent space interpolation for image classification". In: *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE. 2018, pp. 728–733.

[83] Rasool Fakoor, Jonas W Mueller, Nick Erickson, Pratik Chaudhari, and Alexander J Smola. "Fast, accurate, and simple models for tabular data via augmented distillation". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 8671–8681.

[84] Karim Armanious, Chenming Jiang, Marc Fischer, Thomas Küstner, Tobias Hepp, Konstantin Nikolaou, Sergios Gatidis, and Bin Yang. "MedGAN: Medical image translation using GANs". In: *Computerized medical imaging and graphics* 79 (2020), p. 101684.

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

[86] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. “Data Synthesis based on Generative Adversarial Networks”. In: *Proceedings of the VLDB Endowment* 11.10 (2018).

[87] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. “Modeling tabular data using conditional gan”. In: *Advances in Neural Information Processing Systems* 32 (2019).

[88] Juan Manuel Davila Delgado and Lukumon Oyedele. “Deep learning with small datasets: using autoencoders to address limited datasets in construction management”. In: *Applied Soft Computing* 112 (2021), p. 107836.

[89] Toan Tran, Thanh-Toan Do, Ian Reid, and Gustavo Carneiro. “Bayesian generative active deep learning”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6295–6304.

[90] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. “Semi-supervised learning with ladder networks”. In: *Advances in neural information processing systems* 28 (2015).

[91] Laine Samuli and Aila Timo. “Temporal ensembling for semi-supervised learning”. In: *International Conference on Learning Representations (ICLR)*. Vol. 4. 5. 2017, p. 6.

[92] Antti Tarvainen and Harri Valpola. “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results”. In: *Advances in neural information processing systems* 30 (2017).

[93] Vikas Verma, Kenji Kawaguchi, Alex Lamb, Juho Kannala, Arno Solin, Yoshua Bengio, and David Lopez-Paz. “Interpolation consistency training for semi-supervised learning”. In: *Neural Networks* 145 (2022), pp. 90–106.

[94] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. “Mixmatch: A holistic approach to semi-supervised learning”. In: *Advances in neural information processing systems* 32 (2019).

[95] Junpeng Fang, Caizhi Tang, Qing Cui, Feng Zhu, Longfei Li, Jun Zhou, and Wei Zhu. “Semi-Supervised Learning with Data Augmentation for Tabular Data”. In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022, pp. 3928–3932.

[96] Xiaodi Li, Latifur Khan, Mahmoud Zamani, Shamila Wickramasuriya, Kevin W Hamlen, and Bhavani Thuraisingham. “MCoM: A Semi-Supervised Method for Imbalanced Tabular Security Data”. In: *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer. 2022, pp. 48–67.

[97] Sajad Darabi, Shayan Fazeli, Ali Pazoki, Sriram Sankararaman, and Majid Sarrafzadeh. “Contrastive Mixup: Self-and Semi-Supervised learning for Tabular Domain”. In: *arXiv preprint arXiv:2108.12296* (2021).

[98] Talip Ucar, Ehsan Hajiramezanali, and Lindsay Edwards. “Subtab: Subsetting features of tabular data for self-supervised representation learning”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 18853–18865.

[99] Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. “Scarf: Self-Supervised Contrastive Learning using Random Feature Corruption”. In: *International Conference on Learning Representations*. 2022.

[100] Zhifeng Qiu, Wanxin Zeng, Dahua Liao, and Ning Gui. “A-SFS: Semi-supervised feature selection based on multi-task self-supervision”. In: *Knowledge-Based Systems* 252 (2022), p. 109449.

[101] Jennifer Taub, Mark Elliot, Maria Pampaka, and Duncan Smith. “Differential correct attribution probability for synthetic data: an exploration”. In: *International Conference on Privacy in Statistical Databases*. Springer. 2018, pp. 122–137.

[102] Jerome P Reiter. “New approaches to data dissemination: A glimpse into the future (?)” In: *Chance* 17.3 (2004), pp. 11–15.

[103] Bin Yu, Wenjie Mao, Yihan Lv, Chen Zhang, and Yu Xie. “A survey on federated learning in data mining”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 12.1 (2022), e1443.

[104] Kalpana Singh and Lynn Batten. “Aggregating privatized medical data for secure querying applications”. In: *Future Generation Computer Systems* 72 (2017), pp. 250–263.

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

[106] Cynthia Dwork, Aaron Roth, et al. “The algorithmic foundations of differential privacy”. In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.

[107] Kevin Zhang, Neha Patki, and Kalyan Veeramachaneni. “Sequential Models in the Synthetic Data Vault”. In: *arXiv preprint arXiv:2207.14406* (2022).

[108] Yuchao Tao, Ryan McKenna, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. “Benchmarking differentially private synthetic data generation algorithms”. In: *arXiv e-prints* (2021), arXiv–2112.

[109] Adam Kalai and Santosh Vempala. “Efficient algorithms for online decision problems”. In: *Journal of Computer and System Sciences* 71.3 (2005), pp. 291–307.

[110] Aleksandar Nikolov, Kunal Talwar, and Li Zhang. “The geometry of differential privacy: the sparse and approximate cases”. In: *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. 2013, pp. 351–360.

[111] Elizabeth Meckes. “Projections of probability distributions: A measure-theoretic Dvoretzky theorem”. In: *Geometric aspects of functional analysis*. Springer, 2012, pp. 317–326.

[112] Jim Young, Patrick Graham, and Richard Penny. “Using Bayesian networks to create synthetic data”. In: *Journal of Official Statistics* 25.4 (2009), p. 549.

[113] Nicolas Papernot, Martín Abadi, Ùlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. “Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data”. In: *Proceedings of the International Conference on Learning Representations*. 2017. URL: https://arxiv.org/abs/1610.05755.

[114] Martin Benning and Martin Burger. “Modern regularization methods for inverse problems”. In: *Acta Numerica* 27 (2018), pp. 1–111.

[115] Peter L Bartlett, Andrea Montanari, and Alexander Rakhlin. “Deep learning: a statistical viewpoint”. In: *Acta numerica* 30 (2021), pp. 87–201.

[116] Alon Halevy, Peter Norvig, and Fernando Pereira. “The unreasonable effectiveness of data”. In: *IEEE Intelligent Systems* 24.2 (2009), pp. 8–12.

[117] Pedro Domingos. “A few useful things to know about machine learning”. In: *Communications of the ACM* 55.10 (2012), pp. 78–87.

[118] Shaeke Salman and Xiuwen Liu. “Overfitting mechanism and avoidance in deep neural networks”. In: *arXiv preprint arXiv:1901.06566* (2019).

[119] Zeke Xie, Fengxiang He, Shaopeng Fu, Issei Sato, Dacheng Tao, and Masashi Sugiyama. “Artificial neural variability for deep learning: On overfitting, noise memorization, and catastrophic forgetting”. In: *Neural computation* 33.8 (2021), pp. 2163–2192.

[120] Claudio Filipi Gonçalves dos Santos and João Paulo Papa. “Avoiding Overfitting: A Survey on Regularization Methods for Convolutional Neural Networks”. In: *ACM Computing Surveys (CSUR)* (2022).

[121] David A Van Dyk and Xiao-Li Meng. “The art of data augmentation”. In: *Journal of Computational and Graphical Statistics* 10.1 (2001), pp. 1–50.

[122] Sebastien C Wong, Adam Gatt, Victor Stamatescu, and Mark D McDonnell. “Understanding data augmentation for classification: when to warp?” In: *2016 international conference on digital image computing: techniques and applications (DICTA)*. IEEE. 2016, pp. 1–6.

[123] Sima Behpour, Kris M Kitani, and Brian D Ziebart. “Ada: Adversarial data augmentation for object detection”. In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2019, pp. 1243–1252.

[124] David A Van Dyk and Xiao-Li Meng. “The art of data augmentation”. In: *Journal of Computational and Graphical Statistics* 10.1 (2001), pp. 1–50.

[125] Georgios Douzas, Fernando Bacao, Joao Fonseca, and Manvel Khudinyan. “Imbalanced learning in land cover classification: Improving minority classes’ prediction accuracy using the geometric SMOTE algorithm”. In: *Remote Sensing* 11.24 (2019), p. 3040.

[126] Wei Feng, Wenjiang Huang, and Wenxing Bao. “Imbalanced hyperspectral image classification with an adaptive ensemble method based on SMOTE and rotation forest with differentiated sampling rates”. In: *IEEE Geoscience and Remote Sensing Letters* 16.12 (2019), pp. 1879–1883.

[127] Roweida Mohammed, Jumanah Rawashdeh, and Malak Abdullah. “Machine learning with oversampling and undersampling techniques: overview study and experimental results”. In: *2020 11th international conference on information and communication systems (ICICS)*. IEEE. 2020, pp. 243–248.

[128] Julio Hernandez, Jesús Ariel Carrasco-Ochoa, and José Francisco Martínez-Trinidad. “An empirical study of oversampling and undersampling for instance selection methods on imbalance datasets”. In: *Iberoamerican Congress on Pattern Recognition*. Springer. 2013, pp. 262–269.

[129] Bartosz Krawczyk. “Learning from imbalanced data: open challenges and future directions”. In: *Progress in Artificial Intelligence* 5.4 (2016), pp. 221–232.

[130] Teuvo Kohonen. “Emergence of invariant-feature detectors in the adaptive-subspace self-organizing map”. In: *Biological cybernetics* 75.4 (1996), pp. 281–291.

[131] Scott Cost and Steven Salzberg. “A weighted nearest neighbor algorithm for learning with symbolic features”. In: *Machine learning* 10.1 (1993), pp. 57–78.

[132] Augustus Odena, Christopher Olah, and Jonathon Shlens. “Conditional image synthesis with auxiliary classifier gans”. In: *International conference on machine learning*. PMLR. 2017, pp. 2642–2651.

[133] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. “Spatial transformer networks”. In: *Advances in neural information processing systems* 28 (2015).

[134] Timur Sattarov, Dayananda Herurkar, and Jörn Hees. “Explaining Anomalies using Denoising Autoencoders for Financial Tabular Data”. In: *arXiv preprint arXiv:2209.10658* (2022).

[135] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. “Self-supervised learning: Generative or contrastive”. In: *IEEE Transactions on Knowledge and Data Engineering* (2021).

[136] Ehsan Hajiramezanali, Max W Shen, Gabriele Scalia, and Nathaniel Lee Diamant. “STab: Self-supervised Learning for Tabular Data”. In: *NeurIPS 2022 First Table Representation Workshop*. 2022.

[137] Sercan Ö Arik and Tomas Pfister. “Tabnet: Attentive interpretable tabular learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 8. 2021, pp. 6679–6687.

[138] Yue Yu, Jie Chen, Tian Gao, and Mo Yu. “DAG-GNN: DAG structure learning with graph neural networks”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 7154–7163.

[139] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. “Generative adversarial networks”. In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[140] David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. “A learning algorithm for Boltzmann machines”. In: *Cognitive science* 9.1 (1985), pp. 147–169.

[141] Vikram S Chundawat, Ayush K Tarun, Murari Mandal, Mukund Lahoti, and Pratik Narang. “TabSynDex: A Universal Metric for Robust Evaluation of Synthetic Tabular Data”. In: *arXiv preprint arXiv:2207.05295* (2022).

[142] Fida K Dankar, Mahmoud K Ibrahim, and Leila Ismail. “A Multi-Dimensional Evaluation of Synthetic Data Generators”. In: *IEEE Access* 10 (2022), pp. 11147–11158.

[143] Markus Hittmeir, Andreas Ekelhart, and Rudolf Mayer. “On the utility of synthetic data: An empirical evaluation on machine learning tasks”. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*. 2019, pp. 1–6.

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

[145] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. “Improved training of wasserstein gans”. In: *Advances in neural information processing systems* 30 (2017).

[146] Andre Goncalves, Priyadip Ray, Braden Soper, Jennifer Stevens, Linda Coyle, and Ana Paula Sales. “Generation and evaluation of synthetic patient data”. In: *BMC medical research methodology* 20.1 (2020), pp. 1–40.

[147] Mi-Ja Woo, Jerome P Reiter, Anna Oganian, and Alan F Karr. “Global measures of data utility for microdata masked for disclosure limitation”. In: *Journal of Privacy and Confidentiality* 1.1 (2009).

[148] Joshua Snoke, Gillian M Raab, Beata Nowok, Chris Dibben, and Aleksandra Slavkovic. “General and specific utility measures for synthetic data”. In: *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 181.3 (2018), pp. 663–688.

[149] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. “Assessing generative models via precision and recall”. In: *Advances in neural information processing systems* 31 (2018).

[150] Khaled El Emam. “Seven ways to evaluate the utility of synthetic data”. In: *IEEE Security & Privacy* 18.4 (2020), pp. 56–59.

[151] Anat Reiner Benaim, Ronit Almog, Yuri Gorelik, Irit Hochberg, Laila Nassar, Tanya Mashiach, Mogher Khamaisi, Yael Lurie, Zaher S Azzam, Johad Khoury, et al. “Analyzing medical research results based on synthetic data and their relation to real data results: systematic comparison from five observational studies”. In: *JMIR medical informatics* 8.2 (2020), e16492.

[152] Lucas Rosenblatt, Anastasia Holovenko, Taras Rumezhak, Andrii Stadnik, Bernease Herman, Julia Stoyanovich, and Bill Howe. “Epistemic Parity: Reproducibility as an Evaluation Metric for Differential Privacy”. In: *arXiv preprint arXiv:2208.12700* (2022).

[153] Md Sakib Nizam Khan, Niklas Reje, and Sonja Buchegger. “Utility Assessment of Synthetic Data Generation Methods”. In: *Privacy in Statistical Database*. 2022.

[154] Alan F Karr, Christine N Kohnen, Anna Oganian, Jerome P Reiter, and Ashish P Sanil. “A framework for evaluating the utility of data altered to protect confidentiality”. In: *The American Statistician* 60.3 (2006), pp. 224–232.

[155] Tsubasa Takahashi, Shun Takagi, Hajime Ono, and Tatsuya Komatsu. “Differentially Private Variational Autoencoders with Term-wise Gradient Aggregation”. In: *arXiv preprint arXiv:2006.11204* (2020).

[156] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. “Autoaugment: Learning augmentation strategies from data”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 113–123.

[157] Amy Zhao, Guha Balakrishnan, Fredo Durand, John V Guttag, and Adrian V Dalca. “Data augmentation using learned transformations for one-shot medical image segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 8543–8553.

[158] Jing Zhou, Yanan Zheng, Jie Tang, Jian Li, and Zhilin Yang. “Flipda: Effective and robust data augmentation for few-shot learning”. In: *arXiv preprint arXiv:2108.06332* (2021).

[159] Katherina K Hauner, Richard E Zinbarg, and William Revelle. “A latent variable model approach to estimating systematic bias in the oversampling method”. In: *Behavior Research Methods* 46.3 (2014), pp. 786–797.