Synthetic Data, Real Impact

A Framework for Augmenting Tabular Datasets with Synthetic Data in Machine Learning

Jann N. Bitzer

Dissertation written under the supervision of professor

Pedro Afonso Fernandes

Dissertation submitted in partial fulfilment of requirements for the MSc in Business Analytics, at the Universidade Católica Portuguesa, 12.03.2025.

Synthetic Data, Real Impact

A Framework for Augmenting Tabular Datasets with Synthetic Data in Machine Learning

Jann Noah Bitzer

Seminar: Disruptive Forecasting and Business Analytics

Held by: Pedro Afonso Fernandes

Católica Lisbon School of Business & Economics

Católica Lisbon Research Unit in Business & Economics (CUBE)

Católica Lisbon Forecasting Lab (NECEP)

Portugal

March 31, 2025

Abstract

Access to high-quality data is an ever-occurring challenge in machine learning due to scarcity, cost, privacy constraints, and biases. While synthetic data has gained traction in large-scale AI applications to overcome these challenges, its practical implementation for small to mid-size businesses remains underexplored. This study bridges this gap by developing a structured and universal framework to integrate synthetic data augmentation into various machine learning processes. The approach systematically assesses augmentation ratios, selective filtering strategies, and their impact on predictive performance. This research provides a scalable and actionable framework for businesses to use synthetic data, offering practical guidance on augmentation strategies and performance evaluation. By addressing technical and ethical considerations, this study advances the adoption of synthetic data as a transformative tool for data-driven decision-making in business environments.

Keywords: Synthetic Data, Machine Learning, Data Augmentation, Workflow Automation, Business Analytics, Predictive Modeling, AI Implementation.

Title: Synthetic Data, Real Impact - A Framework for Augmenting Tabular Datasets with Synthetic Data in Machine Learning

Author: Jann Bitzer

Resumo

O acesso a dados de alta qualidade é um obstáculo recorrente em aprendizagem automática devido a problemas de escassez de informação, respetivo custo, restrições de privacidade e enviesamento. Embora os dados sintéticos estejam a destacar-se na resolução destes problemas em aplicações de inteligência artificial (IA) em larga escala, a utilização deste tipo de dados por pequenas e médias empresas (PME) não tem sido suficientemente explorada. O presente trabalho de investigação procura colmatar esta lacuna através do desenvolvimento de um processo estruturado e universal que integra dados sintéticos com diferentes métodos de aprendizagem automática. Esta abordagem assenta na avaliação sistemática de diferentes rácios entre dados sintéticos e dados reais, em estratégias de filtragem seletiva e no respetivo impacto na performance preditiva dos modelos. Desta forma, a presente investigação providencia um processo escalável e adaptável a cada negócio que recorre a dados sintéticos, fornecendo orientações práticas em estratégias de aumento da dimensão de bases de dados e de avaliação da performance preditiva. Além disso, a investigação recorre a considerações técnicas e éticas de modo a promover a utilização de dados sintéticos enquanto ferramenta transformadora dos processos de tomada de decisão em meio empresarial.

Palavras-chave: Dados Sintéticos; Aprendizagem Automática; Aumento da Dimensão de Bases de Dados; Automatização de Processos; Analítica de Negócios; Modelação Preditiva; Implementação de Inteligência Artificial.

Título: Dados Sintéticos, Impacto Real – Um Modelo para a Ampliação de Conjuntos de Dados Tabulares com Dados Sintéticos em Machine Learning

Autor: Jann Bitzer

Contents

List of Figures

List of Tables

List of Acronyms

AI Artificial Intelligence

API Application Programming Interfaces

CTGAN Conditional Tabular Generative Adversial Network

CPA Categorical Processing Algorithm

GAN Generative Adversarial Network

LLM Large Language Model

ML Machine Learning

PCA Principal Component Analysis

ROC AUC Receiver Operating Characteristic Area Under the Curve

SDSD Self-Directed Synthetic Dialogues

SDG Synthetic Data Generation

SDV Synthetic Data Vault

Sens Sensitivity

Spec Specificity

TVAE Tabular Variational Autoencoder

VAE Variational Autoencoder

1 Introduction

Companies increasingly rely on data analytics to gain strategic insights and competitive advantages in today's data-driven business landscape. Industries such as entertainment and advertising have demonstrated that proficient data analysis improves decision-making and market success (Davenport, 2005; Ramadan et al., 2020). Building on traditional statistical analysis, Machine Learning (ML) represents the next evolutionary step, enabling businesses to uncover complex patterns and predictions from datasets.

ML research is more popular than ever, dominating submissions on preprint repositories like arXiv, with a significant focus on model training and deployment (Nikolenko, 2021; Klinger et al., 2018). A typical ML pipeline involves collecting raw data, labeling it, training models, and deploying them in real-world applications (Nikolenko, 2021). However, access to high-quality labeled data is a major bottleneck in this process.

High-quality data is essential for innovation and progress across numerous fields. ML models are usually only as good as the data that flows into them. However, access to such data is not always readily available. Many industries contend with datasets that are limited, expensive to acquire, restricted due to privacy regulations, or require extensive time for labeling. This scarcity hinders developing and applying new solutions, slowing innovation and competitive growth. Overcoming these data limitations is crucial for unlocking the full potential of various technologies.

Synthetic Data Generation (SDG) emerges as a promising solution to this problem. Synthesis is commonly used in various physical domains to replicate scarce or costly materials. For example, restaurants often use a horseradish-based substitute to mimic the taste and appearance of authentic wasabi, which is rare and expensive due to its specific growing conditions. The same goes with the textile industry, which uses synthetic dyes to replicate the deep blue color of natural indigo plants, enabling jeans to be dyed efficiently without relying on scarce natural resources (Nikolenko, 2021). Similarly, synthetic data could emulate the statistical properties of real datasets. By generating synthetic data, we could one day entirely overcome the limitations of data scarcity and accessibility, thereby fostering innovation and development across multiple fields.

Although synthetic data has not yet reached a level where it consistently emulates real-world statistical properties with high fidelity, its current state of development already presents valu-

able use cases. This thesis examines how synthetic data in its current state of development could provide businesses with a competitive advantage and help them achieve superior performance. To achieve this objective, the research begins with a comprehensive overview of synthetic data, exploring its theoretical foundations, current methodologies, and practical applications across various industries. This extensive examination sets the stage for understanding how synthetic data can address challenges related to data scarcity, privacy concerns, and labeling inefficiencies. Building upon this foundation, the thesis presents a practical example by developing a SDG framework. This framework demonstrates how organizations can implement synthetic data solutions to potentially overcome data limitations, drive innovation, and improve decision-making processes. By exploring synthetic data's theoretical and practical aspects, the study aims to highlight its significance as a transformative tool for businesses seeking to stay competitive in an increasingly data-driven world.

2 Motivation & Research Questions

Synthetic data is already widely used in developing large-scale Artificial Intelligence (AI) models like a Large Language Model (LLM) (Gunasekar et al., 2023; Liu et al., 2024; Lambert et al., 2024). However, a noticeable gap in the literature concerning SDG exists for companies without substantial venture capital funding to train large-scale models. High-quality data is not just a necessity for major AI corporations; organizations of all sizes rely on it to drive innovation and maintain competitiveness.

The motivation behind this work is the belief that synthetic data can enhance operations and provide a competitive edge significantly, regardless of a company's size, funding, or financial situation. This thesis aims to guide businesses in leveraging SDG for ML algorithms. Furthermore, this research contributes to the democratization of data access. In a world where anyone can synthetically produce the necessary data while adhering to privacy regulations, traditional barriers created by data scarcity and gatekeeping become obsolete. By making synthetic data accessible, smaller enterprises get empowered to innovate and compete on a more level playing field.

Although synthetic data has many applications, this study specifically focuses on its practical application on the augmentation of individual tabular datasets after an extensive exploration of the field. This emphasis is chosen because augmenting single tabular datasets is probably the most prevalent use case among businesses worldwide. Consequently, the practical component of this research is guided by the following research questions:

- RQ1: What are the key parameters and techniques for augmenting tabular datasets with synthetic data to optimize ML model performance?
- RQ2: How can they be systematically integrated into a generalizable augmentation framework?

3 Literature Review

The fields of ML, AI, and SDG are advancing at an unprecedented pace. As such, capturing the current state of the art is challenging. New developments and relevant papers may emerge rapidly, potentially rendering parts of this review outdated shortly after completion. Nevertheless, this literature review aims to synthesize the most significant and recent advancements in SDG, focusing on their applications.

3.1 An Introduction to Synthetic Data

Despite the significant rise in the use of synthetic data over the past few years, there is still no widely accepted definition in scientific literature. The Alan Turing Institute defines synthetic data as *"[...] data that has been generated using a purpose-built mathematical model or algorithm, with the aim of solving a (set of) data science task(s)."* (Jordon et al., 2022). For this work, a new definition will be suggested.

To accurately grasp the concept of synthetic data, it is crucial to distinguish it from data augmentation. Synthetic data is often mistakenly conflated with augmented data—a well-established technique integral to nearly every modern ML pipeline. Data augmentation involves creating artificial variations of existing data points through methods such as rotation, scaling, or noise addition, thereby increasing the diversity of the training dataset and improving model performance.

In contrast, the synthetic data discussed here is not derived from transformations of existing data but involves generating entirely new data points. This approach is particularly valuable in scenarios where real data is costly, scarce, biased, or sensitive concerning privacy. The goal is to generate a dataset that accurately represents what would have been observed if real data were available (Mason et al., 2019). While from-scratch synthetic data may be influenced by the underlying statistical properties of existing data, it does not rely on transforming those data points. Admittedly, the line between sophisticated data augmentation techniques and SDG can be blurry. Therefore, for the purposes of this work, the author proposes the following expanded definition of synthetic data.

Definition 1 *Synthetic data is data that has been created using mathematical models or algorithms designed to learn and replicate the statistical properties of real-world data without*

directly copying it. The objective is to produce a dataset that accurately represents the underlying patterns and structures of real data, thereby enabling data science tasks while addressing issues such as data scarcity, privacy concerns, bias, or accessibility.

The models used in SDG can take various forms, including Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), transformer-based models, diffusion models, statistical simulations, or rule-based algorithms. These models aim to capture the complex relationships within the data to generate new, realistic data points that are valuable for analysis and ML tasks, namely, to train new models.

3.2 Origins of Synthetic Data

The concept of synthetic data has existed for quite some time, with early use cases dating back several decades. Recent advancements in ML and AI have led to more data-intensive models, and coupled with increasing data protection regulations, synthetic data has gained significant traction. Early forms of synthetic data were widely used in research as ground truth. For instance, the usage of synthetic data can be traced back to the 1940s when Stanislaw Ulam and John von Neumann pioneered Monte Carlo simulation methods utilizing synthetic data (Jordon et al., 2022).

In the 1960s, during the nascent stages of ML, particularly in computer vision, synthetic data was essential. Computers at that time lacked the capability to process real-life data, compelling researchers to employ early forms of synthetic data (Nikolenko, 2021). An early example is visual edge detection: Clowes (1971) and Huffman (1971) used analog SDG in the form of handmade drawings to apply edge detection algorithms.

The evolution progressed to computer-generated synthetic data. One of the earliest publications on automatically generated synthetic datasets dates back to 1989 when Pomerleau (1989) trained a neural network for a self-driving car using generated images of roads. Today, training for self-driving cars often occurs in virtual synthetic environments (Leudet et al., 2018). SDG has advanced from analog line drawings to sophisticated generative AI models.

As real-world data sources become scarce, major AI companies are not only investing heavily in SDG to train their foundational models but also accelerating research in this field. Their substantial funding propels advancements, making synthetic data technologies more accessible and effective. In the scientific realm, synthetic data is opening cost-effective avenues for ex-

ploring previously inaccessible domains. The technological research firm Gartner predicts that by 2024, 60% of data used in AI and analytics projects will be synthetically generated, and by 2026, 75% of businesses will employ generative AI to create synthetic customer data—up from less than 5% in 2023. This trend underscores the growing significance of synthetic data in driving innovation and addressing data scarcity across industries (James and Duncan, 2024).

3.3 Synthetic Data Classification

In the literature available today, there is no unified clustering of types of synthetic data that the scientific community agrees on. Some authors categorize by the type of data that is synthetically generated. Most authors, however, use the broader use case of synthetic data instead of the data type. This work only focuses on synthetic data made by generative models and therefore suggests a clustering which is focused on use cases based on Nikolenko (2021):

1. **Fully Synthetic Datasets Generated from Scratch:** This category involves creating entirely synthetic datasets without direct reliance on existing real-world data, aiming to train ML models for real-world application. This approach is particularly beneficial in domains where real data is scarce, expensive, or non-existent. However, a critical challenge lies in ensuring that the synthetic data accurately reflects real-world complexities; failure to do so can result in models that perform poorly when deployed, undermining the very purpose of using synthetic data.

2. **Synthetic Data for Augmenting Existing Datasets:** Here, generative models produce new data points based on existing datasets to create hybrid datasets. Unlike traditional data augmentation—which modifies existing data—this method generates entirely new samples to address biases, enhance diversity, or increase the size of the dataset, thereby potentially improving model performance. Nonetheless, there is a risk of introducing artificial biases or overfitting if the synthetic data does not adequately represent the underlying data distribution or if the generation process is not carefully managed (Raghunathan et al., 2003).

3. **Privacy-Preserving Synthetic Data:** In situations involving sensitive or confidential information, synthetic data can serve as a means to share and analyze data without violating privacy regulations (Raghunathan, 2020). By generating datasets that preserve the statistical properties of the original data, organizations aim to maintain analytical value while

protecting individual identities. However, this approach is not without pitfalls. If the generative models inadvertently memorize and replicate specific details from the original data, there is a risk of re-identification, potentially compromising privacy—a concern that necessitates rigorous validation and the development of robust privacy-preserving techniques.

4. **Synthetic Data Generated from Virtual Simulation Environments:** This category involves the creation of synthetic data through the generation of virtual environments themselves, which are inherently synthetic. These environments—such as physics-based simulations, virtual reality settings, or game engines—are meticulously designed to replicate real-world scenarios and conditions. By simulating environments and interactions, they produce extensive datasets derived from synthetic experiences within these virtual worlds. This approach is particularly valuable in fields like autonomous driving, robotics, manufacturing, and computer vision, where collecting real-world data can be dangerous, expensive, or impractical.

3.4 High Level Limitations of Synthetic Data Methods

While synthetic data offers numerous benefits, it also presents significant challenges and potential pitfalls. The first and most obvious concern is that synthetic data may not accurately capture the complexity and nuances of real-world datasets. If it fails to represent underlying distributions and relationships faithfully, models trained on it may perform poorly in real-world applications (Fu et al., 2024), negating the reason why one should use synthetic data in the first place. Additionally, when not addressed properly, biases inherent in the data generation process can amplify existing biases or introduce new ones, affecting model fairness and reliability.

Furthermore, the misuse of synthetic data can proliferate misinformation. AI models capable of generating human-like data—including text, images, songs, and videos—can be dangerous when used to impersonate real people, manipulate public opinion, or influence political processes (Liu et al., 2024). This misuse erodes trust in legitimate information sources, making it harder to distinguish between truth and falsehood. To mitigate these risks, establishing clear guidelines and robust mechanisms for detecting and countering synthetic misinformation is crucial (Liu et al., 2024).

Synthetic data may also cause ambiguity in AI alignment. Since it might not accurately rep-

resent the complexities of human values and preferences, AI models trained on synthetic data could learn from biased, ungrounded, or misrepresentative data, leading to behaviors misaligned with human expectations (Liu et al., 2024). This ambiguity complicates the interpretation and understanding of AI decision-making processes, making alignment more challenging (Liu et al., 2024).

Finally, generating high-quality synthetic data often requires sophisticated techniques and substantial computational resources, posing practical challenges. Privacy concerns arise as synthetic datasets could potentially be reverse-engineered to reveal details about the original data (Raghunathan, 2020). The lack of established regulatory frameworks and industry standards for synthetic data use also creates legal and ethical uncertainties. These challenges underscore the need for careful consideration and rigorous evaluation when employing synthetic data in ML projects.

Synthetic data is not a universal solution to data scarcity and should be applied with careful consideration. Its use requires a rigorous evaluation of both its effectiveness and ethical implications. A detailed examination of its technical limitations is provided in Chapter 6.

3.5 Synthetic Data in Practice

Synthetic data has emerged as a crucial resource in tackling challenges across numerous fields. While this review highlights some of the most prominent use cases, it does not cover all possible applications nor explore each domain in exhaustive detail. Given the extensive range of areas benefiting from synthetic data, a comprehensive analysis is beyond the scope of this work. Nonetheless, the selected examples are intended to provide a representative overview of the current state of the art in synthetic data utilization across several different domains.

3.5.1 Large-scale models

Synthetic data has become an essential tool in advancing LLMs, addressing challenges like data scarcity, privacy concerns, and high annotation costs (Liu et al., 2024). By artificially generating data that mirrors real-world complexities, researchers can provide LLMs with abundant and diverse training material without the constraints of collecting vast amounts of human-generated content. This approach is particularly beneficial in domains where obtaining real data is difficult or sensitive, such as mathematical reasoning or code generation (Tang et al., 2024; Liu

et al., 2024). For example, synthetic math problems and solutions have been created to enhance LLMs' performance on complex reasoning tasks, enabling models to tackle advanced mathematical concepts beyond their original training scope (Tang et al., 2024). Similarly, synthetic code datasets allow models to learn programming languages and debugging strategies effectively (Liu et al., 2024).

Moreover, synthetic data plays a crucial role in fine-tuning language models to follow instructions and solve complex problems. However, much of the available open data lacks multi-turn conversational data and is often collected using closed models, limiting progress in developing open fine-tuning methods. To address this gap, recent work introduces Self-Directed Synthetic Dialogues (SDSD), an experimental dataset of guided conversations where language models converse with themselves. By incorporating principles from Constitutional AI and creating synthetic preference data, this approach encourages further exploration in multi-turn data and the use of open models to expand the impact of synthetic data (Lambert et al., 2024).

However, utilizing synthetic data also presents challenges in ensuring the accuracy and reliability of the generated content. If not carefully designed, synthetic data can introduce errors or biases, potentially leading models to produce incorrect or untrustworthy outputs. Therefore, while synthetic data significantly contributes to advancing LLM capabilities, it necessitates meticulous generation and validation processes to maintain the models' integrity and applicability in real-world scenarios (Liu et al., 2024).

3.5.2 Computer Vision

Synthetic data has become a key component in advancing computer vision applications. Borkman et al. (2021) discuss the development of the Unity Perception package, which leverages the Unity engine to generate synthetic datasets with high-quality annotations for tasks like object detection and semantic segmentation. This approach enables the creation of diverse and scalable datasets at a lower cost and time compared to manually collected data, with the added advantage of being able to simulate rare events and different environmental conditions, which are crucial for training robust computer vision models (Borkman et al., 2021).

Man and Chahl (2022) further elaborate on the advantages of synthetic image data, such as the ability to automate labeling processes and generate diverse datasets at a lower cost compared to real-world data collection. They also discuss the limitations, like domain gaps and data biases,

which are inherent in synthetic datasets, but suggest that combining synthetic and real datasets can mitigate these issues and enhance model robustness (Man and Chahl, 2022).

3.5.3 Healthcare

Synthetic data is increasingly being employed in healthcare to address critical issues of data accessibility and privacy. According to McDuff et al. (2023), synthetic data enables the creation of datasets that can either complement or fully replace real data in training AI models. This is particularly relevant in healthcare, where privacy regulations and limited access to patient data pose significant challenges.

Synthetic datasets allow for the replication of complex medical conditions and the simulation of various diagnostic scenarios, making them a valuable tool in training models for medical imaging, cardiology, and other specialties while maintaining patient privacy and data security standards (McDuff et al., 2023).

Gonzales et al. (2023) highlight the role of synthetic data in expanding access to health information for research and public health, particularly for testing algorithms and simulating disease outbreaks. Their review identifies key use cases, such as improving predictive analytics, validating ML models, and enabling epidemiological research, showcasing how synthetic datasets provide timely, privacy-preserving solutions to data-sharing challenges in healthcare (Gonzales et al., 2023).

3.5.4 Finance

In the financial sector, synthetic data plays a crucial role in addressing privacy concerns and facilitating the sharing and analysis of sensitive information. Potluru et al. (2023) emphasize the utility of synthetic data in enhancing data access and promoting collaboration without compromising privacy, which is particularly critical in the highly regulated finance industry. They discuss how synthetic data supports various applications, including fraud detection, customer journey analysis, and market counterfactual testing, by enabling counterfactual scenario simulations and augmenting training datasets for ML models (Potluru et al., 2023).

Similarly, Assefa et al. (2020) highlight that synthetic data can alleviate challenges related to the scarcity of historical data for rare events, such as market crashes or structural changes. They outline the role of synthetic data in overcoming internal data-sharing restrictions and addressing

class imbalances in use cases like fraud detection, while ensuring compliance with privacy regulations through different techniques (Assefa et al., 2020).

3.5.5 Synthetic in Business

In recent years, a surge of startups has emerged to meet the growing demand for SDG, offering innovative solutions that balance data utility with privacy concerns. Companies like Mostly AI (Mostly.ai, 2024) and Gretel.ai (gretel.ai, 2024) are at the forefront of this technological advancement.

Mostly AI specializes in creating synthetic datasets that closely mimic real-world data distributions while ensuring individual privacy is preserved. Their platform uses advanced generative models to produce data that retains the statistical properties necessary for accurate analytics and ML tasks without exposing sensitive information. Similarly, Gretel.ai provides developers and organizations with Application Programming Interfacess (APIs) and tools to generate anonymized synthetic data on demand.

By enabling the safe sharing and analysis of data, these startups are instrumental in accelerating research and development across various industries, from healthcare to finance, where data privacy is paramount. Their contributions are not only fostering innovation but also setting new standards for ethical data practices in the age of AI.

4 The Framework

4.1 Introduction

Having established a comprehensive overview of synthetic data, this study practically addresses a broadly relevant problem in data analysis: predictive analytics. Specifically, the focus is on forecasting a target variable Y from a single tabular dataset. By tackling the defined research questions, this chapter presents a detailed, reproducible methodology in the form of a framework for augmenting such datasets, offering both practical guidance and clear documentation. First, an outline of the overall process of synthetic data augmentation for a singular tabular dataset is made. Next, the augmentation procedure will be discussed in detail, emphasizing critical steps and potential pitfalls. Then, a description of the methods used to assess the effectiveness of the augmented data is provided. The goal is first to show the variety of different possibilities how to set up a SDG Pipeline to augment a dataset and then afterwards, in the next chapter, show a simple implementation. To ensure transparency and reproducibility, all synthesizers utilized in this work are openly accessible online, and the Python code used is attached.

4.2 High Level Framework Design

The overall high level experimental workflow begins with an original "raw" dataset, which undergoes rigorous cleaning and preprocessing steps to ensure data quality and consistency. Once a fully preprocessed dataset D_{clean} is obtained, the dataset is split into a training and a holdout dataset. The holdout dataset will be ignored completely throughout the whole augmentation process. The training dataset then is duplicated into two identical subsets. The first subset D_{orig} remains in its processed form without any further modification. The second subset D_{orig} is used to train and fit a chosen synthesizer S. A collection of commonly used synthesizers for tabular data is found in subsection 4.3. The fitting and training of these synthesizers results in a generative model capable of producing synthetic data $D_{syn} = S(D_{orig})$. This synthetic dataset is then appended to D_{orig} to produce an augmented dataset $D_{aug} = D_{orig} \cup D_{syn}$. The process of augmenting the dataset is described in subsection 4.4.
At this point, two datasets are available for further analysis:

1. **Original Dataset**: D_{orig} (cleaned and preprocessed), and

2. **Augmented Dataset**: D_{aug} (cleaned, preprocessed, and augmented with synthetic data).

Both datasets are subsequently used to train identical ML models under the same hyperparameter configurations to predict a target variable Y. Let M_{orig} denote the model trained on D_{orig} and M_{aug} the model trained on D_{aug}. The final step is the evaluation of the augmented dataset trained model with the holdout dataset, benchmarking it against the original dataset. A variety of scores is suggested in subsection 4.5. An overview of the framework is shown in Fig. 1.

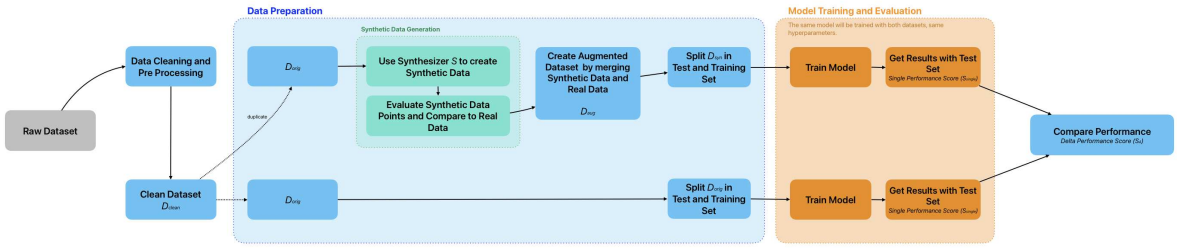


Figure 1: Synthetic Data Generation Framework - High Level Overview

4.3 Synthesizers

The synthetic data used in this study is generated by preconfigured synthesizers. These synthesizers are trained on an original dataset, allowing them to capture its underlying statistical properties and subsequently generate synthetic data. Within this section, four commonly used synthesizers are introduced: the Gaussian Copula Synthesizer, the Conditional Tabular Generative Adversial Network (CTGAN) Synthesizer, the Tabular Variational Autoencoder (TVAE) Synthesizer, and the CopulaGAN Synthesizer. Because the primary focus of this research is on applying these synthesizers rather than on their underlying development, an exhaustive technical exposition of each model's architecture is beyond the scope of this work. Instead, the following overview highlights key characteristics and operational principles of each synthesizer. This serves to convey an understanding of their general functionality, as well as the main differences in their approaches to generating synthetic data.

Selecting the most appropriate synthesizer for a given dataset involves a careful consideration of several interrelated factors. If the dataset exhibits relatively simple distributions with limited feature interactions, a straightforward approach using GaussianCopula may be sufficient.

However, when dealing with complex, non-linear patterns and a mix of numerical and categorical variables, models like CTGAN or TVAE tend to be more effective. Dataset size further influences this choice: small datasets may lead to overfitting with GAN-based methods, making GaussianCopula or TVAE preferable, while large datasets might require strategies to mitigate extended GAN training times, such as optimizing CTGAN's epochs (Miletic and Sariyar, 2024). Moreover, the presence of high-cardinality or imbalanced categorical features can benefit from CTGAN's conditional generation capabilities, though TVAE remains a robust alternative under many conditions. Ultimately, an iterative evaluation—comparing synthetic data quality against real data distributions and downstream model performance—is essential, as no single synthesizer universally excels across all data scenarios.

4.3.1 Gaussian Copula

The process of using a Gaussian copula-based synthesizer, as illustrated in Figure 2 from Patki et al. (2016), begins with a source dataset provided in a standard CSV format. In the case of this work, the csv file will be D_{orig}. This raw table is first enriched by a Categorical Processing Algorithm (CPA) that leverages metadata, such as variable types, semantic annotations, or known constraints, to create an extended representation of the data. With this enriched table as input, the Gaussian copula method is then applied to infer both the individual feature distributions and the joint covariance structure across all variables. By decomposing complex multi-dimensional relationships into simpler, more tractable correlation patterns within a latent Gaussian space, the copula facilitates flexible modeling of dependencies without imposing overly strict parametric assumptions on each marginal distribution. The resulting model captures both the single feature distributions and their covariance patterns, enabling the generation of realistic synthetic datasets that faithfully preserve the statistical properties and inter-variable relationships present in the original data.

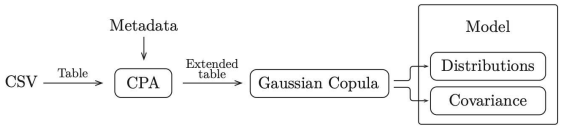


Figure 2: Gaussian Copula Synthesizer Pipeline

4.3.2 CTGAN

While a normal GAN is primarily designed for generating continuous data such as images, videos, or audio by learning their underlying distribution, CTGAN is a deep generative model specifically designed to overcome the challenges of synthesizing tabular data, which often contains a mix of discrete and continuous variables with complex distributions. Traditional GANs struggle with this type of data due to issues such as multi-modal continuous distributions, highly imbalanced categorical data, and the unique combination of discrete and continuous variables. CTGAN introduces key innovations to address these problems, including mode-specific normalization to handle non-Gaussian and multi-modal continuous variables, and a conditional generator with a training-by-sampling approach to mitigate imbalances in categorical columns (Xu et al., 2019). By conditioning the generator on discrete column values and employing a log-frequency sampling strategy during training, CTGAN ensures better representation of all categories while preserving the underlying data distribution. With these features, CTGAN produces high-quality synthetic tabular data.

4.3.3 TVAE

The TVAE Synthesizer is a generative model specifically designed for tabular data. It utilizes a VAE framework, where the encoder maps input data into a compressed latent representation, and the decoder reconstructs the data from this latent space (Xu et al., 2019). The TVAE is particularly suited for mixed-type data (numerical and categorical), as it incorporates mechanisms to handle these complexities effectively. For example, it employs specialized loss functions and embedding techniques to ensure categorical variables are properly represented. By optimizing the latent space to capture the intricate patterns in the original dataset, the TVAE synthesizer generates synthetic data that closely mirrors the original data's statistical properties while allowing for greater flexibility in capturing non-linear and complex relationships. For this work, this synthesizer is an interesting addition for datasets with higher complexities.

4.3.4 CopulaGAN

The CopulaGAN Synthesizer enhances the CTGAN framework by incorporating statistical normalization through copulas before the generative modeling process. It operates in two stages: first, it learns the marginal distributions of each dataset column, identifying their statistical properties (e.g., beta or other parametric distributions), and transforms these into a normalized

Gaussian form with mean $\mu = 0$ and standard deviation $\sigma = 1$. This normalization ensures that the data are standardized, simplifying subsequent modeling. In the second stage, the normalized data are fed into the CTGAN framework, which utilizes GANs to model complex relationships between features, leveraging conditional mechanisms to ensure realistic and consistent SDG (Xu et al., 2019). This combination of copula-based normalization and CTGAN-based learning allows CopulaGAN to capture intricate data patterns while maintaining the statistical fidelity of the original dataset.

4.4 The Augmentation Process

This section zooms in on the actual augmentation process. While the primary objective of SDG is to replicate the statistical properties of real data, the degree to which synthetic data should be incorporated into the training pipeline remains an open question. Excessive augmentation may introduce distortions, leading to performance degradation, whereas insufficient augmentation may fail to produce meaningful improvements. The challenge, therefore, is to determine an optimal augmentation ratio that enhances model generalization without compromising the integrity of learned representations. Besides the augmentation ratio, it is in question if all of the produced synthetic data should be used to augment a dataset. Filtering certain synthetically generated rows can lead to further improvements.

4.4.1 Determining the Appropriate Amount of Synthetic Data

Establishing the appropriate balance between introducing enough synthetic samples to enhance dataset diversity and preventing the inclusion of low-quality or redundant data is a critical initial step. In relatively straightforward settings, where a computationally inexpensive synthesizer such as a Gaussian Copula is paired with a simple model like logistic regression, this balance is often achieved by generating multiple augmentation ratios, training the model on each augmented dataset, comparing performance, and selecting the optimal configuration. However, this scenario rarely reflects practical applications. In most real-world cases, datasets are large, models are complex, and synthesizers are computationally demanding. To address these challenges, this work presents a comprehensive toolbox that streamlines the process of SDG and augmentation for complex, large-scale tasks.

Stratified Sampling for Cost-Effective Data Augmentation A practical strategy for mitigating the high computational costs associated with large-scale datasets and complex synthesizers is to use stratified sampling to create a smaller, yet representative "mini-dataset." By ensuring that each relevant class or subgroup is proportionally included, stratified sampling preserves the distributional characteristics of the original data while substantially reducing its size. This compact dataset can then be augmented at various synthetic data ratios, allowing rapid experimentation and performance assessment without incurring the full computational overhead of training on the entire dataset. Crucially, insights gained from these targeted experiments—such as optimal augmentation ratios and early performance indicators—can be extrapolated to guide larger-scale model training and synthesis efforts. This approach thus serves as an accessible first step in the toolbox for finding an appropriate augmentation ratio and informing subsequent, more resource-intensive investigations.

Resource-Efficient Augmentation Experiments Another valuable strategy for identifying effective augmentation ratios, while managing computational overhead, involves piloting with low-complexity synthesizers and simpler ML models. For instance, Gaussian Copula-based data generation can be paired with a standard logistic regression classifier or an XGBoost Model to form a rather lightweight experimental pipeline. This allows practitioners to iterate rapidly across multiple augmentation ratios, generating performance metrics in a fraction of the time required by more resource-intensive synthesizers or advanced modeling approaches. Such early-phase experiments serve a dual purpose. First, they offer quick indications of the efficacy of different synthetic data proportions. Second, they guide resource allocation and methodological decisions before committing to large-scale, computationally expensive procedures. By systematically refining augmentation ratios on this smaller, more efficient stage, practitioners gain a clearer understanding of which strategies are likely to generalize well, ensuring that subsequent full-scale experiments are more focused and cost-effective.

Large-Scale One-Time Generation for High-Fidelity Augmentation Another valuable approach leverages the final, more powerful synthesizer to generate a large volume of synthetic data in one comprehensive step. By training this advanced model on the original dataset in its entirety, researchers can produce a substantial repository of synthetic samples that can be "sliced" in various proportions to augment the real data. This technique eliminates the need to repeatedly re-run expensive generation processes for each augmentation ratio, thereby reducing

the computational overhead of subsequent experiments. Moreover, it provides a direct window into how the quality of the more sophisticated synthesizer's outputs interacts with the target model, offering detailed insights into the relative benefits of each augmentation level. However, the primary limitation is the significant up-front cost involved in producing such a large synthetic dataset, which can be especially prohibitive when the original dataset is itself extensive or computational resources are constrained.

4.4.2 Selective Filtering of Synthetic Samples

In the preceding chapter, a methodology for generating synthetic data using a variety of ML–based approaches, such as Gaussian Copula, CTGAN, or TVAE was introduced. While synthetic data can substantially expand the training pool and potentially enhance downstream models, not all generated samples are of the same quality. Some may contain implausible values (such as negative ages in a demographic dataset), while others may inadvertently introduce label noise or reinforce existing biases. The objective of this section is to present a structured protocol for refining the synthetic dataset. We first outline domain-specific plausibility checks to remove obviously flawed samples, followed by a more nuanced technique—selective filtering—to further improve the quality and relevance of the generated data.

Preliminary Screening for Data Plausibility An essential initial task involves identifying values that violate fundamental domain constraints. In many application areas, tabular data include numerical fields, such as age, temperature, or income, which all have well-defined ranges. Detecting negative ages in a demographic dataset or impossible sensor measurements in an environmental dataset helps prevent spurious observations from contaminating the training process. Similarly, if categorical fields (for example, country or product category) contain entries outside the recognized taxonomy, these rows should be flagged for correction or removal. To formalize this procedure, let x_j denote the jth feature of a synthetic sample and let $[\ell_j, u_j]$ be the valid range for that feature. Samples for which $x_j \notin [\ell_j, u_j]$ are deemed implausible and excluded.

Further screening involves checking for logical consistency among multiple features. In hospital data, for instance, a discharge date preceding an admission date is inherently contradictory, and domain-specific heuristics should be used to detect such issues. Although generative models often capture basic correlations, outlier cases can still emerge in the synthetic outputs, making it critical to correct or remove any anomalies. Some data points may not be outright invalid

but nonetheless merit additional scrutiny. Extremely large income values, for example, may represent valid outliers in the real domain or artificial artifacts introduced by the generation process. These ambiguities often require either deeper domain expertise or statistical outlier detection techniques. Researchers must evaluate whether to retain, correct, or discard these suspect entries, especially if they may be rare but informative events. Often times synthesizer libraries like the Synthetic Data Vault (SDV) already provide packages to compare the original data versus the synthetically generated one.

Selective Filtering of Synthetic Instances After eliminating obviously flawed observations, the next step is to assess the remaining synthetic samples based on how well they align with the learned decision boundaries of a baseline model trained on real data. This procedure, commonly referred to as selective filtering, aims to keep only those synthetic instances that closely mirror the real data distribution in label space. By reducing label noise and increasing the reliability of augmented data, selective filtering ultimately contributes to more robust predictive models.

The rationale behind selective filtering lies in the observation that generative methods do not always capture class boundaries with perfect fidelity, especially if they are not class-conditional or if class information is only implicitly incorporated. Even class-conditional generators may produce uncertain samples at the edges of these boundaries. To address this issue, a classifier is first trained solely on the original dataset, which ensures that its understanding of the target variable is based on genuine, rather than synthetic, patterns. This baseline model is then used to assign a confidence score to each synthetic instance's predicted label. Instances whose predicted probability for the assigned label is below a chosen threshold (for instance, 0.9) are designated as unreliable and are removed from the augmented pool. This could either be used to filter for elite samples with the risk of amplifying biases of the original dataset, using a number closer to 1. Or eliminate bad samples by setting the threshold rather low.

The process unfolds in stages. First, a suitable baseline classifier, such as logistic regression, a random forest, or XGBoost, is selected and trained on the real dataset. Once the model exhibits sufficiently strong performance on a real-data validation set, each synthetic row is passed through the model to produce a predicted class probability $p(\hat{y} = k \mid \mathbf{x})$, where $\mathbf{x}$ denotes the synthetic features and k the predicted class. If the confidence in this prediction fails to exceed a specified threshold τ, then the row is excluded from further consideration. A grid search or small-scale hyperparameter tuning can be carried out to optimize τ, striking a balance

between retaining sufficient samples and discarding noisy or mislabeled ones.

Trade-Offs and Mitigation Strategies Despite its potential to increase data quality, selective filtering raises two core challenges. First, if the baseline classifier is biased or underrepresents certain groups, it may wrongly reject valuable synthetic samples from those groups. This effect is commonly referred to as bias reinforcement, because the filtered dataset inherits the classifier's inherent biases. One strategy to mitigate this issue is to train multiple filtering models, either on different folds via cross-validation or using diverse modeling techniques, and then aggregate their predictions. By averaging or taking majority votes of several distinct models, the likelihood that one model's biases will dominate the filtering process is reduced.

A second concern involves the loss of edge cases. Rare or borderline samples frequently present low confidence, which makes them susceptible to being filtered out. Yet these very instances can sometimes hold critical insights that improve the robustness of a downstream classifier. To address this limitation, researchers can consider either lowering the threshold for specific minority classes or retaining a buffer of moderately low-confidence synthetic samples for further analysis. Employing a separate validation process to examine whether these marginal instances significantly benefit performance can inform decisions about how conservative or inclusive the filtering should be.

A related caution pertains to overfitting or self-reinforcing feedback. If the same model architecture or data partitioning scheme is employed both for generating and filtering the synthetic data, there is a risk of reinforcing the same learned patterns. Training a filtering model on a distinct subset or employing a model architecture different from that used in the generative stage can reduce the propensity for such overfitting.

In Summary The techniques described in this chapter serve as a two-tiered protocol for refining synthetic datasets prior to augmenting them with real data. Initial plausibility checks ensure that invalid or contradictory observations, such as out-of-range numerical values or logically inconsistent cross-feature combinations, are detected and removed. Selective filtering then leverages a well-trained baseline classifier to eliminate synthetic instances that diverge significantly from established decision boundaries, thereby reducing the risk of label noise. Although both steps can substantially bolster the utility and fidelity of the synthetic dataset, they must be undertaken with an awareness of the inherent trade-offs, especially the potential for bias

amplification or the inadvertent removal of informative edge cases.

4.5 Evaluating Performance

To evaluate the influence of synthetic data augmentation, common performance metrics such as Sensitivity (Sens) and Specificity (Spec) are computed for both models as Delta Performance Scores (S_Δ). Sensitivity (also called recall) measures the proportion of actual positives correctly identified by the model, while specificity quantifies the proportion of actual negatives correctly classified. We define the improvement metrics as the differences in performance between the augmented and non-augmented models, for example:

$$\Delta\text{Sens} = \text{Sens}(M_{aug}) - \text{Sens}(M_{orig})$$
$$\Delta\text{Spec} = \text{Spec}(M_{aug}) - \text{Spec}(M_{orig})$$

A positive ΔSens or ΔSpec indicates that the augmented dataset provides improved model performance compared to the original dataset. Higher values of these differences suggest that the chosen synthesizer and augmentation strategy are beneficial. This entire experimental procedure, including the flow from raw data preprocessing to model training and final evaluation, is illustrated schematically in Fig. 1.

To asses further assess the augmented datasets, a number of metrics is suggested. Concretely, the following categories of performance scores will be derived:

1. **Single Performance Scores (S_{single}):** For each combination of dataset, synthesizer, and metric, two performance values will be measured—one for the original dataset D_{orig} and one for the augmented dataset D_{aug}.

2. **Delta Performance Scores (S_Δ):** Building on the single performance scores, a set of difference values (i.e., augmentation gains) will be computed to quantify the performance changes attributed to the synthetic augmentation. For each dataset-synthesizer-metric triplet, the delta score ΔS is defined as the difference between augmented and original performance: $\Delta S = S_{aug} - S_{orig}$.

3. **Synthesizer Performance Scores (S_{syn}):** When comparing different synthesizers, gaining insight into the effectiveness of each synthesizer independently, the delta scores associated with a given synthesizer will be aggregated across all datasets and metrics. This results in a higher-level assessment of synthesizer quality.

In summary, the overarching aim is to determine whether the proposed approach leads to measurable improvements in model performance. The expanded experimental design provides a structured hierarchy of performance metrics, enabling a nuanced analysis of how synthetic data augmentation influences ML outcomes. Furthermore, this framework facilitates direct comparisons across multiple synthesizers, thereby guiding the selection of the most effective synthesizer for a given task.

5 Framework in Action

Within this chapter, an application of the framework is demonstrated, encompassing various techniques for synthetic data augmentation with the objective of enhancing machine learning model performance. Due to the limited scope of this thesis, only a basic augmentation pipeline is employed using the most computationally efficient synthesizer and basic augmentation techniques, rather than exploring the full spectrum of techniques. Although extensive preprocessing and rigorous hyperparameter tuning are known to improve performance significantly, the approach is intentionally limited to basic preprocessing without specific hyperparameter tuning to highlight the performance differences between the synthetically augmented dataset and the original dataset.

5.1 The example Dataset

The used example dataset contains 284,807 credit card transactions that occurred over a two-day period in September 2013, involving European cardholders (Gianluca Bontempi, 2020). The dataset has a total of 30 features and one target variable. Due to confidentiality, 28 out of the 30 features are the outcome of a Principal Component Analysis (PCA) transformation. These features are labeled V1 - V28. Two features have not been transformed with PCA, time and amount. While the feature time represents the seconds elapsed between each transaction and the first transaction in the dataset, amount represents the transaction amount. As the target variable, each transaction is labeled either as legitimate or as fraudulent.

The dataset is highly imbalanced, comprising 99.83% legitimate transactions and 0.17% frauds (492 fraudulent transactions in total). The low fraud rate is underscoring both the rarity of fraud events and the challenge of designing reliable predictive models.

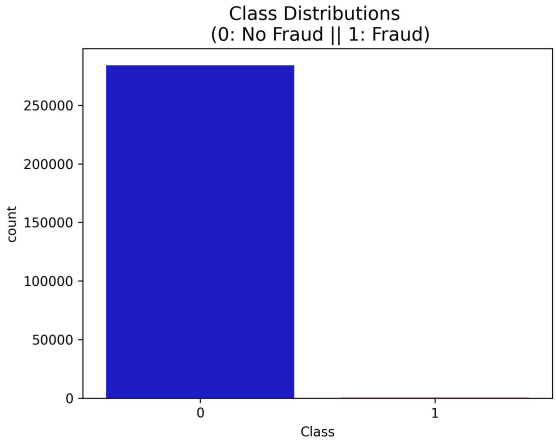
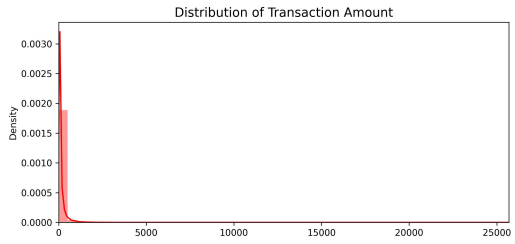
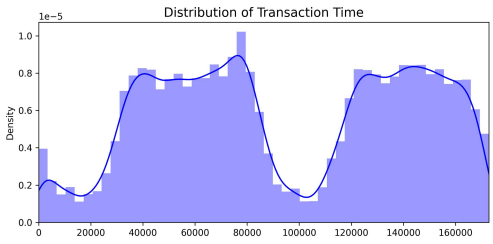


Figure 3: Target Variable Distribution

Figure 4 presents a comparative analysis of transaction distributions, specifically focusing on transaction amount and timing. Plot (a) illustrates the distribution of transaction amounts, revealing a pronounced skewness toward smaller values, indicating that the majority of transactions involve relatively low amounts, with larger amounts occurring infrequently. In contrast, plot (b) demonstrates the distribution of transactions over time, highlighting two clear peaks that suggest significant temporal patterns or periodicity in transaction activity. These temporal peaks could indicate higher transactional activity during specific periods, possibly corresponding to a daily cycle.



(a) Transaction Amount Distribution



(b) Distributions of Transactions over Time

Figure 4: Comparison of Transaction Distributions

Since Figure 4a visually only shows that most of the transaction amounts are rather small, let's dive deeper into the transaction amount by using a box plot with a log transformation. Figure 5 illustrates the distribution of transaction amounts in the dataset. The data exhibits significant skewness with numerous high-value outliers. While the presence of such extreme values might suggest data entry errors or anomalies, in the context of fraud detection, high-value

transactions can indeed constitute important evidence of fraudulent activity. Therefore, these outliers were intentionally preserved to ensure that potentially critical fraud signals are retained for analysis. To mitigate the potential impact of these extreme values on model performance, an XGBoost classifier, which is known for robustness against skewed data distributions and resilience to outliers due to its tree-based architecture, was selected.

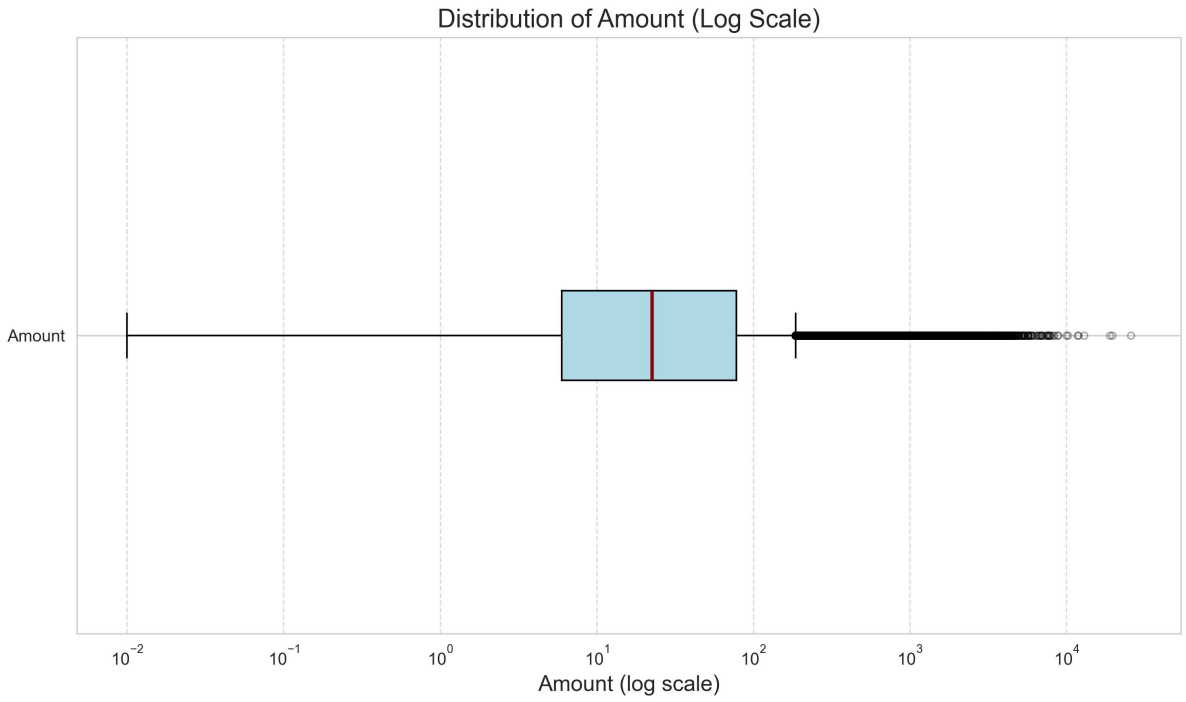


Figure 5: Distribution of Transaction Amount (Log Scale)

Figure 6 presents side-by-side box plots of the 28 principal components (V1 through V28), providing a consolidated view of their central tendencies, overall spread, and potential extreme values. A few components (for example, V2 and V20) exhibit comparatively wider interquartile ranges and a higher number of outliers, suggesting these latent dimensions might capture greater variability in underlying behavioral or transaction-level characteristics. In contrast, components such as V16 and V22 maintain a more compact distribution, implying that they encode more homogeneous aspects of the data. The presence of some outliers across multiple components raises considerations for downstream modeling, particularly in credit card fraud detection settings where extreme observations often signal abnormal (and potentially fraudulent) patterns. Methods such as robust scaling or transformation may be warranted to prevent these outliers from exerting disproportionate influence, yet domain expertise can help distinguish between legitimate, high-value transactions and spurious anomalies. Due to the usage of an XGBoost

model and reasons described above, for now, no further actions on the outliers were taken. However, the patterns observed in Figure 5 underscore the need for careful treatment of extreme values to preserve valuable information while minimizing distortion in subsequent analyses.

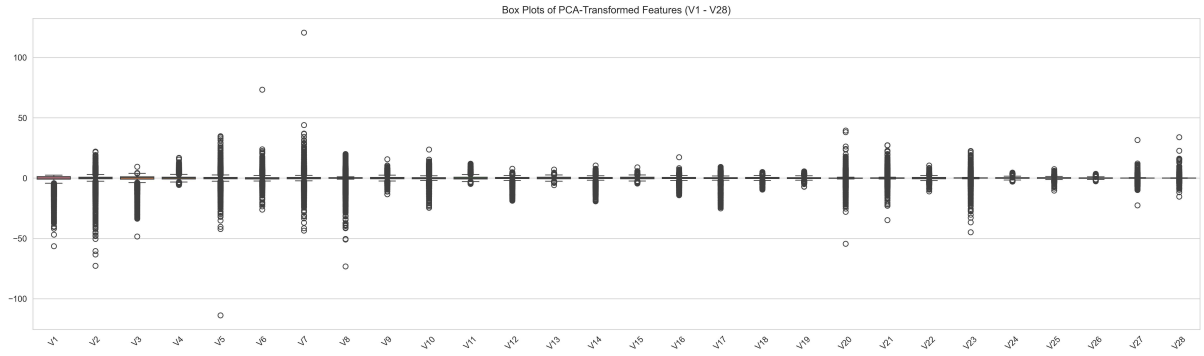


Figure 6: Boxplot Features V1-28

The correlation heatmap presented in Figure 7 illustrates pairwise correlations between the variables in the dataset. Overall, the majority of variables exhibit low correlations, with absolute values mostly below 0.5, indicating weak linear relationships. Notably, moderate correlations are identified between 'Amount' and variables V2 (-0.53), V5 (-0.39), and V7 (0.22), suggesting some degree of linear dependency among these features. Such moderate correlations could potentially lead to multicollinearity, which can affect the stability and interpretability of certain statistical models, particularly regression-based methods. However, given the generally low correlations observed, severe multicollinearity is unlikely to be a substantial issue for this dataset.

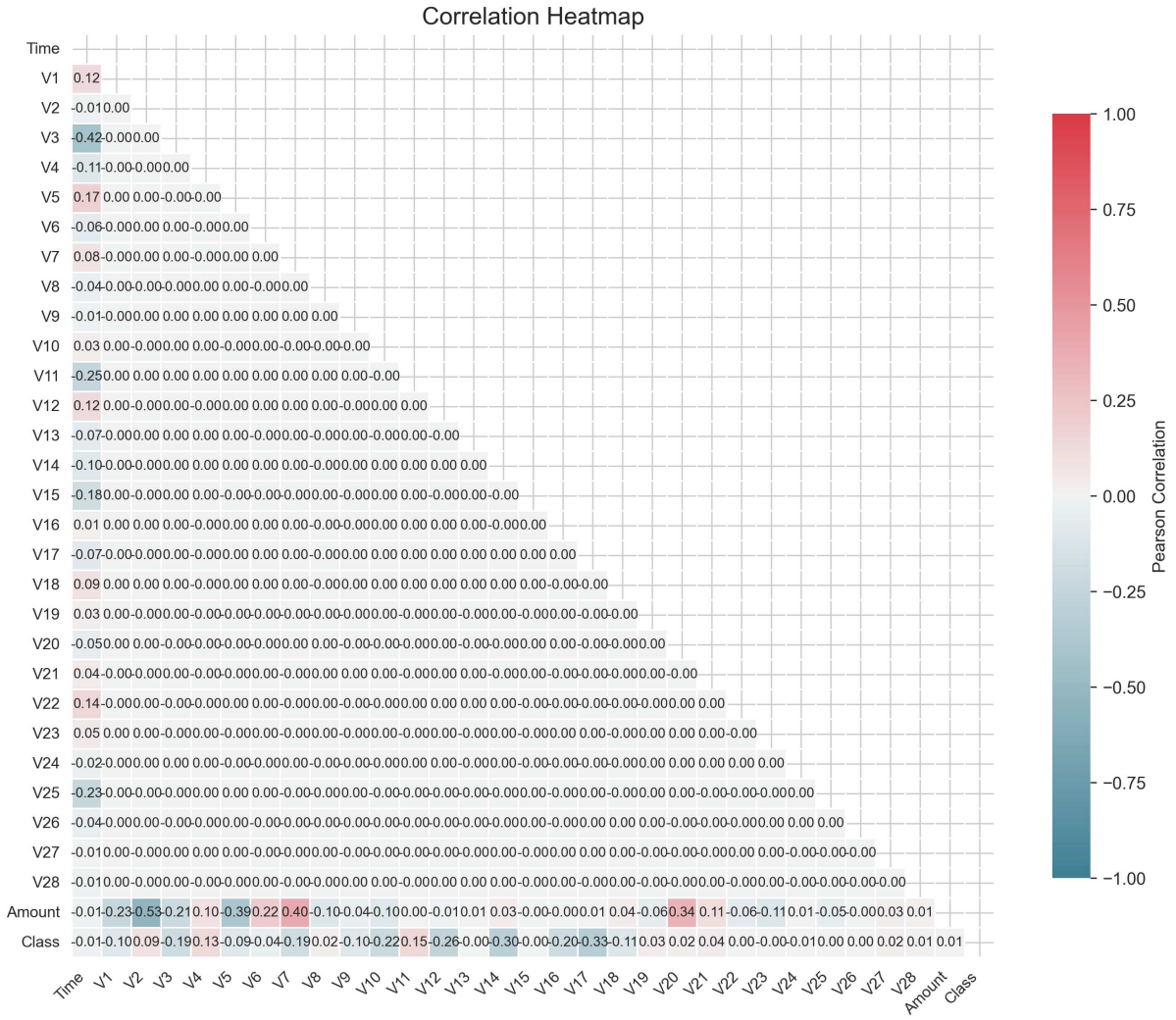


Figure 7: Correlation Heatmap

Taken together, the selected dataset comprises 284,807 credit card transactions from European cardholders over a two-day period, characterized by high class imbalance (99.83% legitimate transactions and 0.17% fraud cases). Exploratory data analysis highlights the skewed nature of transaction amounts, significant temporal patterns, and notable extreme values. Principal components analysis reveals variability across latent features, with some components exhibiting considerable spread and outliers indicative of behavioral anomalies. Correlation analysis identifies moderate linear relationships between transaction amount and select principal components but suggests limited risk of severe multicollinearity. Given these insights and the decision to employ an XGBoost classifier, known for robustness against skewed distributions and outliers, further preprocessing was deemed unnecessary at this stage to preserve critical fraud signals and due to the already high quality of the chosen dataset. Still, when observing results, these choices should be remembered.

5.2 Procedure

Following exploratory data analysis and preprocessing, a holdout set is separated from the original dataset for subsequent evaluation purposes. This holdout set remains completely independent and is excluded from the training of data synthesizers. The remaining dataset, henceforth referred to as *Dorg*, serves as the basis for synthetic data generation.

The synthetic augmentation process begins by investigating how varying ratios of synthetic data influence classification performance, according to Chapter 4.4. Then, a Gaussian Copula synthesizer is fitted on *Dorg* to capture the underlying statistical characteristics of the original data. Subsequently, synthetic datasets are generated at ten distinct augmentation ratios relative to: 0.1, 0.2, 0.3, 0.4, 0.8, 1, 2, 4, 8, and 16 times its original size, creating multiple synthetic datasets.

A XGBoost classifier is then trained on each synthetic dataset, with hyperparameters consistently tuned across all iterations to ensure comparability. Model performance is assessed using six metrics highly relevant to fraud detection tasks and scenarios with class imbalance: Accuracy, Precision, Recall (Sensitivity), F1 Score, Specificity, and the Receiver Operating Characteristic Area Under the Curve (ROC AUC). Evaluation is conducted using the previously reserved holdout set.

Based on the evaluation outcomes across these metrics, an optimal augmentation ratio is identified. The synthetic data corresponding to this optimal ratio is further examined in detail, comparing it against the real data to detect potential anomalies or irregularities. If necessary, selective filtering procedures are implemented and evaluated. Upon confirmation of beneficial results, selective filtering is applied to refine the synthetic dataset.

Finally, the filtered synthetic data is combined with the original training dataset *Dorg* to form an augmented training dataset, while maintaining the initial holdout set for validation. XGBoost classifiers are retrained on both the original and augmented datasets, and the models' performance is comprehensively assessed following the criteria outlined in Chapter 4.5.

5.3 Results

The results are divided into two subsections. The first subsection investigates the optimal proportion of synthetic data for augmenting the original dataset and examines whether selectively incorporating high-quality synthetic samples improves model performance. The second sub-

section presents the model performance results following the prescribed workflow, including benchmark comparisons to evaluate the effectiveness of synthetic data augmentation relative to the original dataset. Both sections use the example dataset described above.

5.3.1 Augmentation Process Results

Testing Number of Rows Results Table 1 shows the model performance of the 10 synthetically generated datasets. The comprehensive results indicate that Accuracy remained exceptionally high (99.95%) for almost all augmentation ratios, reaching its maximum value of 0.99963 at 0.8x. While this figure appears impressive, it should be interpreted with caution because of the substantial class imbalance, which can lead to elevated Accuracy when the majority class is much larger than the minority class.

Precision showed a distinct improvement at an augmentation ratio of 0.8x, peaking at 0.95294, which suggests that moderate levels of synthetic data can reduce the false-positive rate more effectively. Recall, on the other hand, was maximized (0.83673) at a 4x augmentation ratio, indicating that higher ratios of synthetic data helped the model capture more fraudulent transactions. The F1 Score, which balances Precision and Recall, was highest at 0.8x augmentation (0.88525), demonstrating that this intermediate ratio offered a beneficial trade-off between capturing positive cases and maintaining low false alarms.

Specificity was consistently very high (0.99980) across all augmentation levels, reflecting the relative ease with which legitimate (negative) transactions were correctly identified. ROC AUC values also remained robust, exceeding 0.95 for all tested ratios, with a marginal peak of 0.98709 at 0.3x augmentation. Despite these variations, performance remained strong across a wide range of augmentation levels, although 0.8x exhibited an especially favorable balance across multiple metrics.

These findings underscore the utility of synthetic data augmentation for addressing the challenges associated with a highly imbalanced credit card dataset, particularly in improving the detection of fraudulent (minority) cases. The generally high Accuracy and Specificity mirror the dominance of legitimate transactions; however, closer inspection of Precision, Recall, and F1 Score is necessary to understand how well fraudulent cases are being identified.

Augmentation at 0.8× produced especially compelling outcomes, offering the highest F1 Score and a notable peak in Precision. This balance suggests that 0.8× augmentation provides a practi-

Table 1: Performance Metrics for Different Augmentation Ratios

Ratio	Accuracy	Sensitivity (Recall)	Specificity	F1 Score	Precision	ROC AUC
0.1	0.99954	0.78571	0.99991	0.85556	0.93902	0.97938
0.2	0.99961	0.82653	0.99991	0.88043	0.94186	0.98555
0.3	0.99956	0.81633	0.99988	0.86486	0.91954	0.98709
0.4	0.99954	0.80612	0.99988	0.85870	0.91860	0.97503
0.8	0.99963	0.82653	0.99993	0.88525	0.95294	0.97726
1	0.99956	0.79592	0.99991	0.86188	0.93976	0.97736
2	0.99956	0.79592	0.99991	0.86188	0.93976	0.96684
4	0.99958	0.83673	0.99986	0.87234	0.91111	0.96483
8	0.99958	0.79592	0.99993	0.86667	0.95122	0.96880
16	0.99958	0.81633	0.99989	0.86957	0.93023	0.97666
Dorg	0.99954	0.80612	0.99988	0.85870	0.91860	0.97432

cal middle ground: it supplies enough synthetic data to bolster minority-class representation and improve fraud detection without introducing potential pitfalls of large-scale synthetic oversampling, such as overfitting to artificial patterns. In risk-averse settings where the highest possible Recall is essential, a larger augmentation ratio (e.g., 4×) may be more suitable, although it also requires greater computational resources and may reduce overall Precision.

In conclusion, while performance remains high across various augmentation levels, the results point to 0.8× as an attractive option for many credit card fraud detection scenarios. Practitioners should ultimately weigh these performance considerations against deployment constraints, including computational overhead and the risk tolerance for false positives versus false negatives, to select the most appropriate augmentation strategy.

Optimizing the generated Data Due to the performance of the different augmentation ratios, the ratio of 0.8 was chosen as the dataset to pursue with. Following the screening procedures described above, the synthetic dataset for basic domain validity, outlier presence, and logical consistency was evaluated. First, a simple range check to each numerical variable x_j was applied. For instance, variables that represent strictly non-negative quantities (e.g., the "Amount" field) were verified to ensure the synthetic generator had not produced negative values. In this

dataset, no instances of negative amounts were detected, suggesting the model had effectively learned the underlying domain constraints. Similarly, for the time-indexed feature ("Time"), all generated values fell within plausible temporal ranges, aligning with those observed in the original dataset.

Next, the empirical distributions of corresponding features between the real and the synthetic data were compared. Visual inspection of the density plots for features V_1 through V_28 indicates that, in most cases, the synthetic model captured the central tendency and the general shape of each feature's distribution reasonably well. Examples therefore are visually shown in Fig. 8 and Fig. 9.

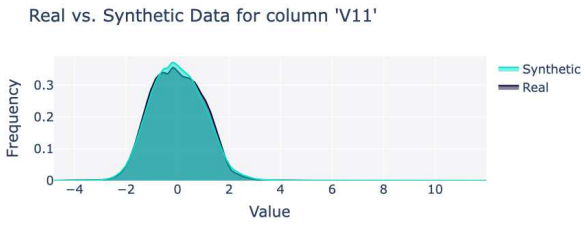


Figure 8: Feature V11 Synthetic Generated Data vs. Real Data Distribution

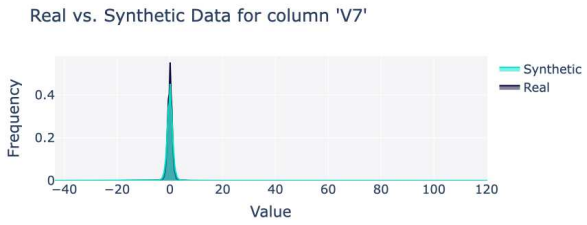


Figure 9: Feature V7 Synthetic Generated Data vs. Real Data Distribution

However, subtle deviations become apparent, especially in the tails. Especially the features V_2 and V_24 exhibit heavier or lighter tails in the synthetic data than in the real data (Fig. 10 and Fig. 11), raising the possibility that the generative model either attenuated or exaggerated extreme values.

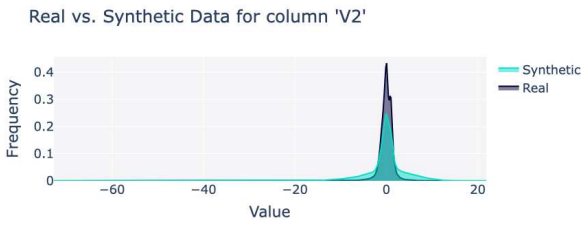


Figure 10: Feature V2 Synthetic Generated Data vs. Real Data Distribution

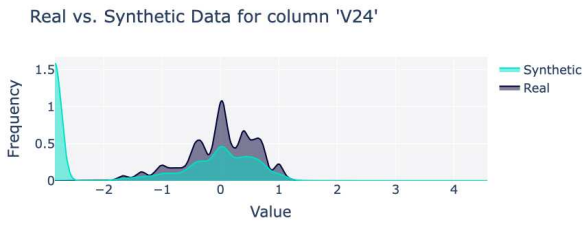


Figure 11: Feature V24 Synthetic Generated Data vs. Real Data Distribution

Finally, we examined the class balance in real vs. synthetic data. As expected, the original data displayed a highly imbalanced class distribution (Fig. 12), whereas the synthetic model, depending on its training configuration, could reproduce a similarly skewed distribution or over-sample the minority class. Here, the synthetic data maintained a near-identical distribution of

fraud vs. non-fraud cases, indicating that the model did not artificially alter the target proportion.

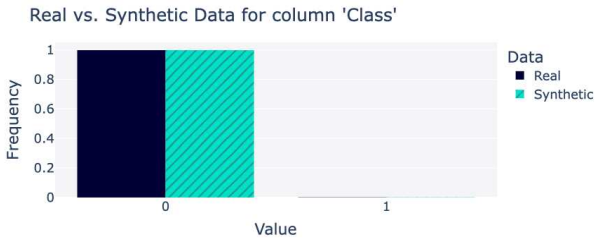


Figure 12: Target Variable Synthetic Generated Data vs. Real Data Distribution

5.3.2 Performance Scores

Table 2 presents a side-by-side comparison of the baseline model, trained only on the original dataset, versus the model enhanced by synthetic data augmentation. Following the evaluation logic of chapter 4.5, three different scores were calculated: Single Performance Scores, Delta Performance Scores as well as a Synthesizer Performance Score. When it comes to the Single Performance Scores, accuracy for the baseline model is 0.9995, increasing to 0.9996 in the augmented model. Precision moves from 0.9186 in the original setup to 0.9529 with augmentation, while recall (sensitivity) increases from 0.8061 to 0.8265. The F1 score changes from 0.8587 to 0.8852, and ROC AUC shifts from 0.9743 to 0.9773. Specificity remains at 0.9999, showing a marginal change in the thousandths place. Therefore the Delta Performance Scores range from 0.0001 to 0.0343. The final row of the table reports a Synthetic Performance Score of 0.1048, which aggregates the individual delta values across metrics.

Table 2: Comparison of Original vs. Augmented Dataset Performance

Metric	**Original Data**	**Augmented Data**	**Delta Score**
accuracy	0.9995	0.9996	0.0001
sensitivity (recall)	0.8061	0.8265	0.0204
specificity	0.9999	0.9999	0.0001
f1	0.8587	0.8852	0.0266
precision	0.9186	0.9529	0.0343
roc_auc	0.9743	0.9773	0.0029
Synth. Performance Score			0.1048

5.4 Discussion

Testing Number of Rows Results The findings suggest that a moderate level of synthetic data augmentation (specifically, around 0.8×) offers a strong trade-off among precision, recall, and F1 score while maintaining high accuracy and specificity. Although nearly all augmentation ratios preserve elevated accuracy—largely a consequence of the dominant legitimate class—closer inspection of recall and precision reveals that 0.8× augmentation strikes a particularly favorable balance, boosting minority-class detection without substantially increasing the false-positive rate. By contrast, higher augmentation ratios (e.g., 4×) may further boost recall but risk diminished precision and greater computational overhead. These results highlight the importance of aligning augmentation strategies with specific risk-management needs (e.g., tolerance for false positives versus false negatives) and available resources.

Optimizing generated Data Overall, the plausibility checks and distribution comparisons suggest that the generative model learned a majority of the underlying data patterns with reasonable fidelity. The close alignment in most univariate distributions indicates success in capturing the core statistical properties for each feature. A small fraction of implausible or outlying entries nonetheless underscores the importance of domain-level screening. Even well-trained generative models can produce extreme or inconsistent values—particularly in the lower-density regions of the feature space—thus routine application of range checks and correlation assessments remains essential.

Decisions about whether to retain, correct, or remove flagged points should be guided by domain expertise. For example, extremely large "Amount" values might be legitimate outliers in financial transactions, whereas nonsensical correlations (e.g., a negative or nonsensical "Time" shift) may reveal generative model artifacts that offer no real analytical value. In practice, iterative refinements—retraining the synthesizer or adjusting the model's hyperparameters—can reduce the frequency of implausible points in later generations. Moreover, while comparing synthetic and real distributions is crucial, purely statistical alignment does not guarantee that subtle semantic or logical constraints are preserved, highlighting the need for ongoing collaboration with domain experts. These screening and discussion steps thus ensure that synthetic data augmentation not only boosts sample size and diversity but also maintains a high level of realism and consistency for subsequent modeling tasks.

Evaluating Performance The systematic improvement across nearly all metrics suggests that the inclusion of synthetic samples positively influenced the classification performance. The rise in accuracy, although slight, shows that the model consistently recognizes the correct label more often when supplemented with augmented data. More notably, the gains in recall (sensitivity) and precision signify a reduction in both missed detections and false alarms. This leads to a corresponding increase in F1, indicating a more balanced capture of minority-class events alongside improved precision.

Meanwhile, the shift in ROC AUC from 0.9743 to 0.9773 indicates that the augmented model maintains a favorable trade-off between true positive rate and false positive rate across multiple thresholds. Specificity, already near a ceiling, remains stable, demonstrating that the additional synthetic data did not undermine performance on the majority class. Finally, the overall Synthetic Performance Score of 0.1048 suggests that, collectively, the metric improvements attained through synthetic augmentation are substantive, underscoring the potential value of generating and incorporating synthetic samples for enhanced model robustness. However, the aggregated Synthetic Performance Score is mostly important when comparing the performance of different synthesizers. In this example, there is only one synthesizer. Therefore, besides confirming a positive impact of synthetic data augmentation, the score has no relevance in this scenario.

Discussion of Potential Enhancements While the present findings confirm the feasibility of synthetic data augmentation with only minimal constraints, there is ample scope for further optimization by adopting more advanced techniques. Harnessing sophisticated GANs or VAE, for instance, may capture nuanced data distributions with higher fidelity, especially for under-represented classes. By systematically balancing the minority-class generation process, these methods could bolster recall and reduce false negatives, ensuring that important but infrequent patterns are not overlooked. A second avenue for improvement involves selective filtering, whereby synthetic samples are screened through domain-informed criteria or statistical outlier detection, weeding out entries that deviate too sharply from real-world plausibility. This filtering would not only preserve the diversity introduced by synthetic examples but also help prevent model drift caused by unrepresentative artifacts. Third, synthetic data generation in this case could also be used to upsample the minority class. In this case, training the synthesizer only on fraudulent cases and producing synthetic data with it could increase, for example, Sensitivity. Finally, the evaluation of the synthetic data could be tailored to the specific domain. When

calculating the Synthesizer Performance Score, each metric could get a weight assigned. In the case of the fraud detection with a specific focus on detecting true positives, precision could be assigned a higher weight than the other metrics.

Summary The findings presented in this work illustrate the substantial promise of synthetic data augmentation. Even without employing selective filtering or applying strict domain constraints, integrating synthesized samples consistently elevated performance across key metrics. This outcome suggests that even the simplest implementation of augmentation, with a rather simple synthesizer, can assist in capturing minority-class variations, mitigating underrepresentation, and improving the model's overall discriminative power.

At the same time, these results should be viewed as a preliminary indication of what is achievable. More sophisticated augmentation strategies—incorporating domain-specific knowledge, refined outlier detection, or strategic selective filtering—could yield additional gains in both accuracy and interpretability. By demonstrating how straightforward synthesis methods can already bolster minority-class detection, this study offers a compelling glimpse into the untapped potential of synthetic data. Future work that systematically integrates targeted filtering rules, deeper statistical checks, and tighter alignment with expert knowledge stands poised to unlock even greater improvements, thereby paving the way for more robust and reliable applications in diverse real-world settings.

6 Practical Limitations

Though synthetic data has gained considerable attention and this work advocates its use across a variety of domains, it is by no means a universal remedy for every dataset that struggles with performance or suffers from limited sample sizes. While synthetic data can indeed be a powerful tool for augmenting scarce or imbalanced datasets, there is no guarantee that it will always lead to improvements or even make things worse. Its effectiveness hinges on numerous factors, such as the fidelity of the generative model, the complexity of the real-world phenomena under study, and the suitability of the synthetic samples for the specific learning task. In what follows, key limitations that practitioners should bear in mind when considering synthetic data for their ML workflows will be identified and discussed.

Data Quality and Representativeness One fundamental issue in generating synthetic samples is ensuring that these artificial observations truly reflect the underlying patterns and distributions of the original dataset. The ultimate goal when generating synthetic data to train a ML model with it should always be to boost the performance when used with real data. Any mismatch between the real and synthetic distributions, such as overfitting to noise, underfitting minority subgroups, or producing unrealistic outliers, can introduce biases that negatively impact subsequent model training. This risk is especially pronounced when the real dataset is small or highly imbalanced, limiting the generative model's ability to capture nuanced relationships. If synthetic data fails to reflect relevant feature correlations or complex interactions accurately, the augmented dataset could mislead the learning algorithm rather than enhance it, resulting in reduced generalizability and potential fairness concerns. In this work's practical part, the synthesizer was not able to learn the correct distribution of some features. This was uncovered by comparing the real data to the synthetically generated one—a crucial step to ensure a close approximation of the real data.

Generalization vs. Overfitting in SDG SDG methods, whether based on GANs, VAEs, or more classical sampling approaches, require careful tuning to avoid overfitting. Overfitting as in "memorized" data points that are too close to real observations, thus failing to provide truly novel samples. Alternatively, an underfit model may produce samples lacking the variation or richness of the real data. Both extremes risk undermining performance improvements. Achieving a balanced fit often involves computationally expensive hyperparameter optimiza-

tion, a deep understanding of the domain, and avoiding data leakage. The insights in this work, therefore, provide the tools to deal with these risks.

Evaluation Challenges Assessing the quality of synthetic tabular data poses unique challenges. Unlike images or text, where artifacts can be spotted through visual or linguistic inspection, tabular data may be high-dimensional, multi-modal, and less interpretable without specialized metrics. Standard measures, like likelihood estimates, discriminative accuracy of an auxiliary classifier, or distribution-distance metrics, frequently fail to capture more nuanced structural dependencies among features. Moreover, there is no universal yardstick to determine whether synthetic data is "good enough" for a given application. Consequently, the process of validating synthetic data remains somewhat subjective and context-dependent, complicating reproducibility and cross-study comparisons. Therefore, this work mainly introduced performance measures at the end of the ML pipeline to learn about the quality by evaluating the results.

Domain Dependence and Lack of Theoretical Guarantees The effectiveness of synthetic data augmentation can vary widely across domains and data types. Certain domains (e.g., sensor data, controlled manufacturing processes) may exhibit relatively simple statistical behaviors, making them more amenable to synthetic augmentation. Others (e.g., healthcare or financial datasets) can include complex, non-linear interactions and heterogeneous sub-populations that are difficult to model. The absence of strong theoretical guarantees on performance gains in diverse settings restricts the applicability of a single approach. Each use case may demand domain-specific feature engineering, generative modeling techniques, and exhaustive validation to ensure meaningful improvements in model performance. There is no one approach that fits all.

Scalability and Computational Constraints Training state-of-the-art generative models can be computationally demanding, especially for large and high-dimensional tabular datasets. GANs, VAEs, or ensemble-based methods often require extensive trial-and-error hyperparameter tuning, resulting in high computational overhead. Hardware constraints, such as limited GPU memory, may force practitioners to reduce model size or training iterations, potentially compromising the fidelity of the generated data. This trade-off between computational feasibility and data quality can limit how frequently or extensively synthetic augmentation is applied, espe-

cially in rapidly evolving environments where repeated, real-time updates of synthetic datasets may be desirable.

Optimal Ratio of Synthetic to Real Data A central challenge, addressed in Chapter 4.4, remains in determining the optimal proportion of synthetic data to real data for model training. While introducing synthetic data can improve model generalization, excessive augmentation may dilute the signal from real data or introduce synthetic biases that deteriorate performance. There is no universal guideline on the ideal augmentation ratio, as it is highly dependent on dataset characteristics, model complexity, and task-specific requirements. This study explored augmentation strategies; however, further research is needed to develop principled methodologies for dynamically adjusting the synthetic-to-real data ratio based on empirical performance metrics. Without such frameworks, practitioners must rely on heuristic approaches, which may not generalize well across different datasets and applications.

Interpretability and Transparency Many generative methods, particularly deep learning approaches, function as black boxes that lack clear interpretability. Understanding how synthetic data points are created, what real data points most heavily influenced the generative process, and how distortions arise is crucial for high-stakes applications. In fields like healthcare or finance, stakeholders often demand explicit explanations for the data's origins and fidelity. Merely stating that generated samples are "statistically similar" may not suffice. This opacity can impede the adoption of synthetic data augmentation strategies, as decision-makers may be uncomfortable relying on data that lack an auditable lineage or interpretable rationale.

In summary, while synthetic data augmentation presents an attractive route to addressing challenges such as data scarcity, cost of data, class imbalance, and privacy concerns, the above limitations underscore the complexities involved. Future research should pursue more robust and transparent generative methodologies, standardized evaluation metrics, and stronger theoretical frameworks to guide practitioners in effectively integrating synthetic data into real-world ML pipelines. Equally important is the need to manage ethical and privacy implications, ensuring that synthetic data enhances model performance without compromising fairness or individual protections.

7 Conclusion

By showcasing a practical framework for synthetic data in a business context and its application, this study contributes to the growing body of literature on data augmentation and its implications for predictive analytics. The motivation behind this research comes from the critical challenge of data scarcity, which continues to impede innovation and model accuracy across multiple domains. While synthetic data has in the recent past been heavily leveraged in large-scale AI models, this work extends its application to businesses and organizations that lack extensive resources and funding. The results demonstrate that strategically generated synthetic data can mitigate constraints associated with limited or expensive real-world observations, offering a viable solution to data accessibility challenges in order to increase the performance of machine learning models.

While this work at times dreams of a world without data scarcity and unlimited access to high quality data, it also manages to give this dream a reality check. Despite its promising implications, the research acknowledges several limitations warrant further exploration. The fidelity of synthetic data remains a key concern, as generative models may introduce artifacts or fail to capture complex real-world interactions. Moreover, the computational overhead associated with advanced generative techniques, such as CTGAN and TVAE, presents scalability challenges that must be addressed before widespread adoption. Additionally, the ethical and regulatory considerations surrounding synthetic data usage, particularly in privacy-sensitive domains such as healthcare and finance, necessitate robust governance frameworks to prevent unintended biases or potential re-identification risks that would enable trace backs to the original data.

Future research should explore several critical directions to build upon the findings of this study. First, developing hybrid generative models that integrate multiple synthesis techniques could enhance the representational power of synthetic data. Second, more sophisticated evaluation metrics should be devised to comprehensively assess the utility of synthetic data beyond traditional performance indicators. Third, domain-specific adaptations of synthetic data techniques could improve their applicability across industries with unique data characteristics and constraints. Lastly, continued advancements in privacy-preserving generative models, such as differentially private synthetic data, could further enhance the trustworthiness of synthetic augmentation strategies.

In conclusion, this thesis provides a robust foundation for understanding and implementing syn-

thetic data augmentation in ML pipelines. While challenges remain, the findings reinforce the potential of synthetic data as a key enabler of data-driven innovation. As organizations continue to grapple with data scarcity and privacy concerns, synthetic data offers a compelling alternative to traditional data acquisition methods, paving the way for more inclusive and scalable AI solutions. By advancing our understanding of SDG and its strategic deployment, this research contributes to the broader discourse on how ML can be democratized, ensuring that data-driven advancements remain accessible and impactful across diverse sectors.

References

S. Assefa, D. Dervovic, M. Mahfouz, T. Balch, P. Reddy, and M. Veloso. Generating synthetic data in finance: opportunities, challenges and pitfalls. Technical report, JP Morgan Chase, 7 2020. URL https://ssrn.com/abstract=3634235.

S. Borkman, A. Crespi, S. Dhakad, S. Ganguly, J. Hogins, Y.-C. Jhang, M. Kamalzadeh, B. Li, S. Leal, P. Parisi, C. Romero, W. Smith, A. Thaman, S. Warren, and N. Yadav. Unity Perception: Generate Synthetic Data for Computer Vision. 7 2021. URL http://arxiv.org/abs/2107.04259.

M. B. Clowes. On Seeing Things. Technical report, 1971.

T. H. Davenport. Competing on Analytics. Technical report, Harvard Business Review, 2005. URL www.hbrreprints.org.

F. Fu, J. Chen, J. Zhang, C. Yang, L. Ma, and Y. Yang. Are Synthetic Time-series Data Really not as Good as Real Data? Technical report, 2 2024.

Gianluca Bontempi. DEFEATFRAUD: Assessment and validation of deep feature engineering and learning solutions for fraud detection, 3 2020.

A. Gonzales, G. Guruswamy, and S. R. Smith. Synthetic data in health care: A narrative review. *PLOS Digital Health*, 2(1), 1 2023. ISSN 27673170. doi: 10.1371/journal.pdig.0000082.

gretel.ai. gretel.ai Website, 2024. URL https://gretel.ai.

S. Gunasekar, Y. Zhang, J. Aneja, C. C. T. Mendes, A. Del Giorno, S. Gopi, M. Javaheripi, P. Kauffmann, G. de Rosa, O. Saarikivi, A. Salim, S. Shah, H. S. Behl, X. Wang, S. Bubeck, R. Eldan, A. T. Kalai, Y. T. Lee, and Y. Li. Textbooks Are All You Need. 6 2023. URL http://arxiv.org/abs/2306.11644.

D. A. Huffman. Impossible objects nonsense-sentence. 2 1971.

S. James and A. D. Duncan. Over 100 Data, Analytics and AI Predictions Through 2030 Gartner for Data & Analytics Leaders. Technical report, Gartner, 6 2024.

J. Jordon, L. Szpruch, F. Houssiau, M. Bottarelli, G. Cherubin, C. Maple, S. N. Cohen, and A. Weller. Synthetic Data – what, why and how? 5 2022. URL http://arxiv.org/abs/2205.03257.

J. Klinger, K. Stathoulopoulos, and J. Mateos-Garcia. Is AI eating software? An analysis of AI/ML research trends using scientific pre-prints, 2018.

N. Lambert, H. Schoelkopf, A. Gokaslan, L. Soldaini, V. Pyatkin, and L. Castricato. Self-Directed Synthetic Dialogues and Revisions Technical Report. 7 2024. URL http://arxiv.org/abs/2407.18421.

J. Leudet, T. Mikkonen, F. Christophe, and T. Männistö. Virtual environment for training autonomous vehicles. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10965 LNAI, pages 159–169. Springer Verlag, 2018. ISBN 9783319967271. doi: 10.1007/978-3-319-96728-8{_}14.

R. Liu, J. Wei, F. Liu, C. Si, Y. Zhang, J. Rao, S. Zheng, D. Peng, D. Yang, D. Zhou, and A. M. Dai. Best Practices and Lessons Learned on Synthetic Data. 4 2024. URL http://arxiv.org/abs/2404.07503.

K. Man and J. Chahl. A Review of Synthetic Image Data and Its Use in Computer Vision, 11 2022. ISSN 2313433X.

K. Mason, S. Vejdan, and S. Grijalva. An "On The Fly" Framework for Efficiently Generating Synthetic Big Data Sets. 3 2019. URL http://arxiv.org/abs/1903.06798.

D. McDuff, T. Curran, and A. Kadambi. Synthetic Data in Healthcare. 4 2023. URL http://arxiv.org/abs/2304.03243.

M. Miletic and M. Sariyar. Challenges of Using Synthetic Data Generation Methods for Tabular Microdata. *Applied Sciences (Switzerland)*, 14(14), 7 2024. ISSN 20763417. doi: 10.3390/app14145975.

Mostly.ai. Mostly.ai Website, 2024. URL https://mostly.ai.

S. I. Nikolenko. Synthetic Data for Deep Learning. Technical report, 6 2021. URL http://www.springer.com/series/7393.

N. Patki, R. Wedge, and K. Veeramachaneni. The synthetic data vault. In *Proceedings - 3rd IEEE International Conference on Data Science and Advanced Analytics, DSAA 2016*,

pages 399–410. Institute of Electrical and Electronics Engineers Inc., 12 2016. ISBN 9781509052066. doi: 10.1109/DSAA.2016.49.

D. A. Pomerleau. ALVINN: An Automonous Land Vehicle in a Neural Network. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 1*, pages 305–313. Morgan Kaufmann Publishers Inc., 1989.

V. K. Potluru, D. Borrajo, A. Coletta, N. Dalmasso, Y. El-Laham, E. Fons, M. Ghassemi, S. Gopalakrishnan, V. Gosai, E. Kreačić, G. Mani, S. Obitayo, D. Paramanand, N. Raman, M. Solonin, S. Sood, S. Vyetrenko, H. Zhu, M. Veloso, and T. Balch. Synthetic Data Applications in Finance. 12 2023. URL http://arxiv.org/abs/2401.00081.

T. E. Raghunathan. Annual Review of Statistics and Its Application Synthetic Data. *Annual Review ofStatistics and Its Application*, 9 2020. doi: 10.1146/annurev-statistics-040720. URL https://doi.org/10.1146/annurev-statistics-040720-.

T. E. Raghunathan, J. P. Reiter, and D. B. Rubin. Multiple Imputation for Statistical Disclosure Limitation. Technical Report 1, 2003. URL https://ecommons.cornell.edu/server/api/core/bitstreams/2be137aa-4b6a-454f-be26-e041e8b79e49/content.

M. Ramadan, H. Shuqqo, L. Qtaishat, H. Asmar, and B. Salah. Sustainable competitive advantage driven by big data analytics and innovation. *Applied Sciences (Switzerland)*, 10(19), 10 2020. ISSN 20763417. doi: 10.3390/app10196784.

Z. Tang, X. Zhang, B. Wang, and F. Wei. MathScale: Scaling Instruction Tuning for Mathematical Reasoning. 3 2024. URL http://arxiv.org/abs/2403.02884.

L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni. Modeling Tabular data using Conditional GAN. 6 2019. URL http://arxiv.org/abs/1907.00503.

Appendix

March 11, 2025

[1]:
```
#basic
import numpy as np
import pandas as pd

#matplotlib
import matplotlib.pyplot as plt

#sklean
from sklearn import metrics
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression, LinearRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn import model_selection
from sklearn.utils import class_weight
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
 ↪f1_score, confusion_matrix, roc_auc_score
from sklearn.utils import resample
from sklearn.pipeline import Pipeline

#imblearn
from imblearn.over_sampling import SMOTE

#xgboost
import xgboost as xgb
from xgboost import XGBClassifier, plot_importance

#sdv
from sdv.single_table import GaussianCopulaSynthesizer
from sdv.metadata import Metadata
```

```
from sdv.evaluation.single_table import get_column_plot

#seaborn
import seaborn as sns

#warnings
import warnings
```

0.1 1. Load Cleaned and Preprocessed Dataset

[2]:
```
#load cleaned and preprocessed dataset
data = pd.read_csv('creditcard.csv')
```

[3]:
```
#quick dataset check
data.head()
```

[3]:
```
   Time        V1        V2        V3        V4        V5        V6        V7  \
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         V8        V9  …       V21       V22       V23       V24       V25  \
0  0.098698  0.363787  … -0.018307  0.277838 -0.110474  0.066928  0.128539
1  0.085102 -0.255425  … -0.225775 -0.638672  0.101288 -0.339846  0.167170
2  0.247676 -1.514654  …  0.247998  0.771679  0.909412 -0.689281 -0.327642
3  0.377436 -1.387024  … -0.108300  0.005274 -0.190321 -1.175575  0.647376
4 -0.270533  0.817739  … -0.009431  0.798278 -0.137458  0.141267 -0.206010

        V26       V27       V28  Amount  Class
0 -0.189115  0.133558 -0.021053  149.62      0
1  0.125895 -0.008983  0.014724    2.69      0
2 -0.139097 -0.055353 -0.059752  378.66      0
3 -0.221929  0.062723  0.061458  123.50      0
4  0.502292  0.219422  0.215153   69.99      0

[5 rows x 31 columns]
```

[4]:
```
data.shape
```

[4]:
```
(284807, 31)
```

[5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
```

```
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

[6]:
```
print(data['Class'].value_counts())
```

```
Class
0    284315
1       492
Name: count, dtype: int64
```

0.2 2. Create Holdout Set

[7]:
```
#create holdout set
train_data, holdout_set = train_test_split(
    data,
    test_size=0.2,
    random_state=42,
    stratify=data['Class']
)

print("Taining Data Value Count:")
print(train_data['Class'].value_counts())
print("_______________________")
print("Holdout Set Value Count:")
print(holdout_set['Class'].value_counts())
```

```
Taining Data Value Count:
Class
0    227451
1       394
Name: count, dtype: int64
-----------------------
Holdout Set Value Count:
Class
0    56864
1       98
Name: count, dtype: int64
```

0.3 3. The Augmentation Process

0.3.1 3.1 Create Synthetic Datasets of different size

[29]:
```
#find out the right augmentation ratio

# Suppress specific warnings
warnings.filterwarnings("ignore", category=UserWarning)

#create dictionary to save results
#synthetic_datasets = {}

# Function to generate synthetic datasets
def generate_synthetic_datasets(data_set, augmentation_ratios):

    # Prepare the synthesizer
    metadata = Metadata.detect_from_dataframe(data=data_set)
    synthesizer = GaussianCopulaSynthesizer(metadata)
    synthesizer.fit(data_set)
```

```
    #Generate and store synthetic datasets for each ratio
    synthetic_datasets = {}
    for ratio in augmentation_ratios:
        syn_rows = int(ratio * len(data_set))
        synthetic_sample = synthesizer.sample(num_rows=syn_rows)

        # Combine synthetic data with the original sample
        combined_data = pd.concat([data_set, synthetic_sample],␣
 ↪ignore_index=True)

        synthetic_datasets[ratio] = combined_data

        # Print class distribution if desired
        print(f"Ratio {ratio}:")
        print(combined_data['Class'].value_counts())
        print()

    return synthetic_datasets

# Define specific values for nr_rows_per
augmentation_ratios = [0.1, 0.2, 0.3, 0.4, 0.8, 1, 2, 4, 8, 16]

# Generate synthetic datasets
synthetic_datasets = generate_synthetic_datasets(train_data,␣
 ↪augmentation_ratios)

synthetic_datasets["Dorg"] = train_data
#synthetic_datasets["mini"] = mini_set
```

```
Ratio 0.1:
Class
0    250183
1       446
Name: count, dtype: int64

Ratio 0.2:
Class
0    272955
1       459
Name: count, dtype: int64

Ratio 0.3:
Class
0    295680
1       518
Name: count, dtype: int64
```

```
Ratio 0.4:
Class
0    318434
1       549
Name: count, dtype: int64

Ratio 0.8:
Class
0    409401
1       720
Name: count, dtype: int64

Ratio 1:
Class
0    454889
1       801
Name: count, dtype: int64

Ratio 2:
Class
0    682355
1      1180
Name: count, dtype: int64

Ratio 4:
Class
0    1137256
1       1969
Name: count, dtype: int64

Ratio 8:
Class
0    2047060
1       3545
Name: count, dtype: int64

Ratio 16:
Class
0    3866630
1       6735
Name: count, dtype: int64
```

0.3.2 3.2 Model and Test all Augmentation ratios

[33]:
```
target_variable = 'Class'

def model_u_results(data_set, holdout_data):

    # --- Define target and explanatory variables ---
    X_train = data_set.drop(columns=[target_variable])
    y_train = data_set[target_variable].values
    X_test = holdout_data.drop(columns=[target_variable])
    y_test = holdout_data[target_variable].values

    # --- Define XGBoost model ---
    model = xgb.XGBClassifier(use_label_encoder=False,
                              eval_metric='logloss',
                              random_state=11)

    # Train
    model.fit(X_train, y_train)

    # Predictions (labels)
    y_pred = model.predict(X_test)

    # Predictions (probabilities) - for additional metrics
    y_prob = model.predict_proba(X_test)[:, 1]

    # Compute metrics
    accuracy = accuracy_score(y_test, y_pred)
    tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()
    sensitivity = tp / (tp + fn)  # same as recall
    specificity = tn / (tn + fp)
    f1 = f1_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    roc_auc = roc_auc_score(y_test, y_prob)

    results_df = pd.DataFrame([{
        'Model': 'XGBoost',
        'Accuracy': accuracy,
        'Sensitivity': sensitivity,
        'Specificity': specificity,
        'F1 Score': f1,
        'Recall': recall,
        'Precision': precision,
        'ROC AUC': roc_auc
    }])
```

```
    return results_df
```

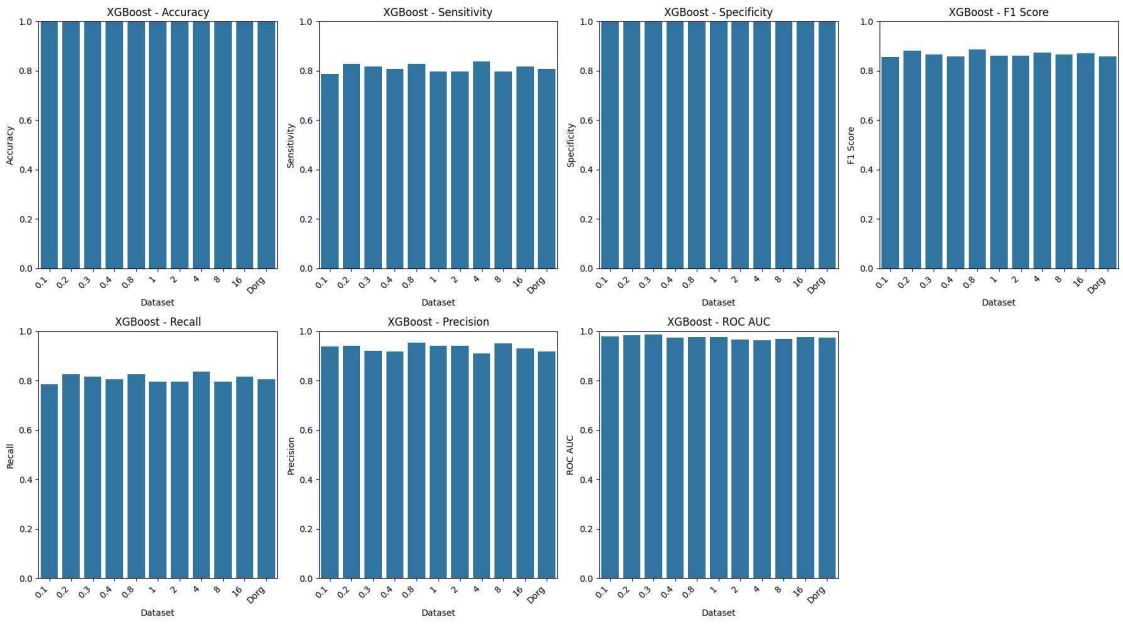
[31]:
```
model_results = []

for data_set in synthetic_datasets:
    results = model_u_results(synthetic_datasets[data_set], holdout_set)
    model_results.append((data_set, results))
    print(str(data_set) + " -> done")
```

```
0.1 -> done
0.2 -> done
0.3 -> done
0.4 -> done
0.8 -> done
1 -> done
2 -> done
4 -> done
8 -> done
16 -> done
Dorg -> done
```

0.3.3 3.3 Compare Synthetic Datasets Performance

[32]:
```
all_results = []
for ds_name, df in model_results:
    temp_df = df.copy()
    temp_df['Dataset'] = ds_name  # Mark which dataset these rows belong to
    all_results.append(temp_df)

all_results_df = pd.concat(all_results, ignore_index=True)

# Visualization: One figure per model, one subplot per metric
metrics = ["Accuracy", "Sensitivity", "Specificity",
           "F1 Score", "Recall", "Precision", "ROC AUC"]

# Unique model names (e.g., "Logistic Regression", "Random Forest", etc.)
models = all_results_df["Model"].unique()

for model in models:
    # Filter the big DF to just this model's rows
    model_df = all_results_df[all_results_df["Model"] == model]

    # Create a figure with 2 rows x 4 cols => 8 subplots
    fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(18, 10))
    axes = axes.ravel()  # Flatten the axes array for easy iteration

    for i, metric in enumerate(metrics):
```

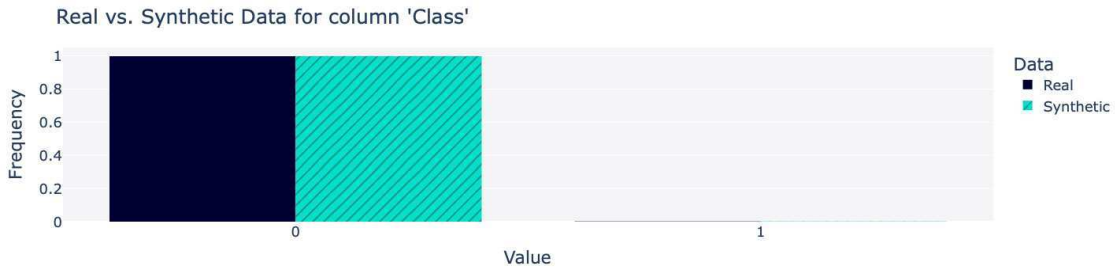
```
        ax = axes[i]
        sns.barplot(
            data=model_df,
            x="Dataset",
            y=metric,
            ax=ax
        )
        ax.set_title(f"{model} - {metric}")
        ax.set_xticklabels(ax.get_xticklabels(), rotation=45, ha="right")

        # Set the Y-axis limit from 0.8 to 1
        ax.set_ylim(0.0, 1.0)

    # If fewer than 8 metrics, hide the extra subplots
    if len(metrics) < len(axes):
        for j in range(len(metrics), len(axes)):
            axes[j].set_visible(False)

    plt.tight_layout()
    plt.show()
```



0.3.4 3.4 Analyzing the Synthetic Data created with a selected ratio

[14]:
```
metadata = Metadata.detect_from_dataframe(data=synthetic_datasets["Dorg"])
```

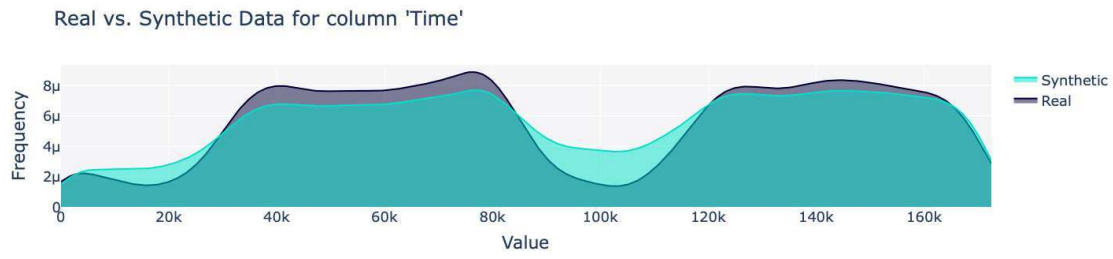
[15]:
```
fig = get_column_plot(
    real_data=synthetic_datasets["Dorg"],
    synthetic_data=synthetic_datasets[0.8],
    column_name='Class',
    metadata=metadata
)

fig.show()
```

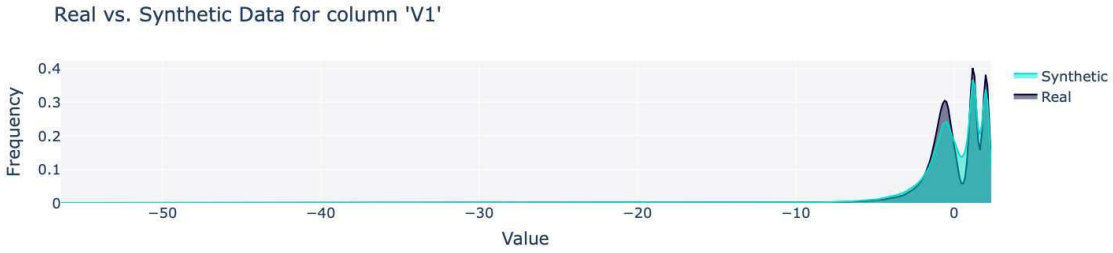


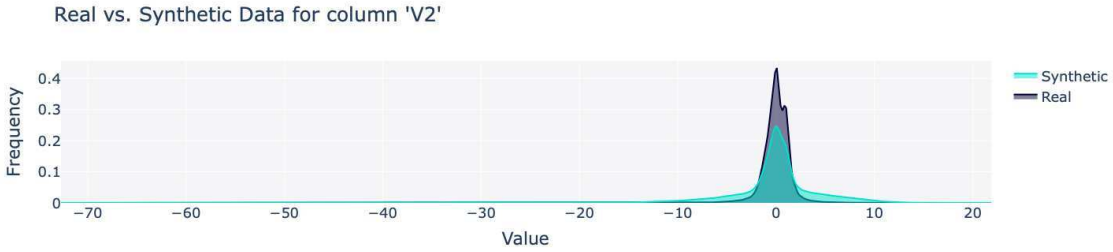
[16]:
```
negative_count = (synthetic_datasets[0.8]["Time"] < 0).sum()

if negative_count > 0:
    print(f"There are {negative_count} rows with 'Time' below 0.")
else:
    print("No rows with 'Time' below 0.")
```

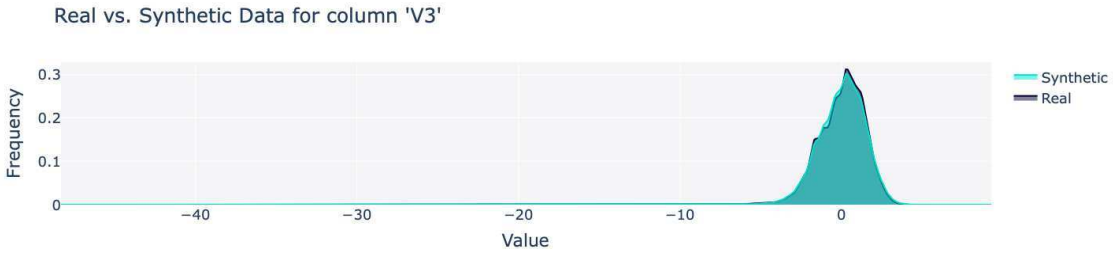
```
No rows with 'Time' below 0.
```

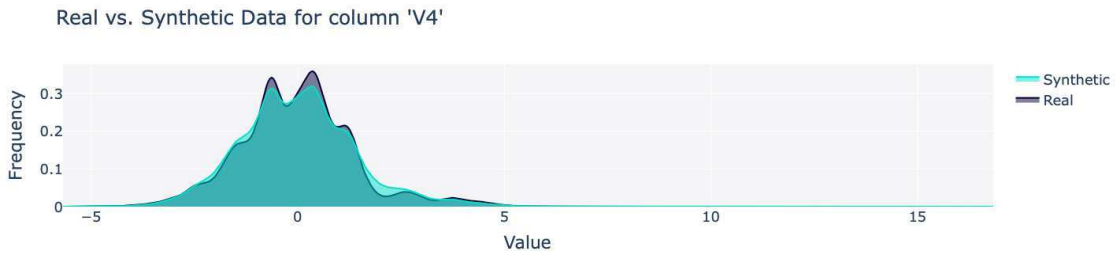
[17]:
```
columns = synthetic_datasets["Dorg"].columns

for column in columns:
    fig = get_column_plot(
    real_data=synthetic_datasets["Dorg"],
    synthetic_data=synthetic_datasets[0.8],
    column_name=column,
    metadata=metadata
    )

    fig.show()
```

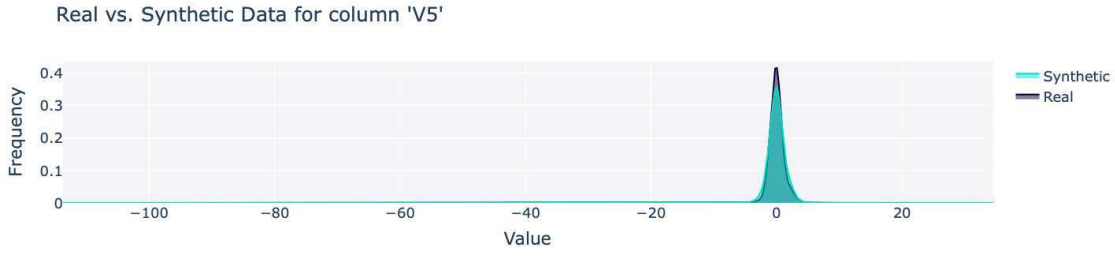

Real vs. Synthetic Data for column 'Time'
Frequency
8μ
6μ
4μ
2μ
0
0
20k
40k
60k
80k
100k
120k
140k
160k
Value
Synthetic
Real

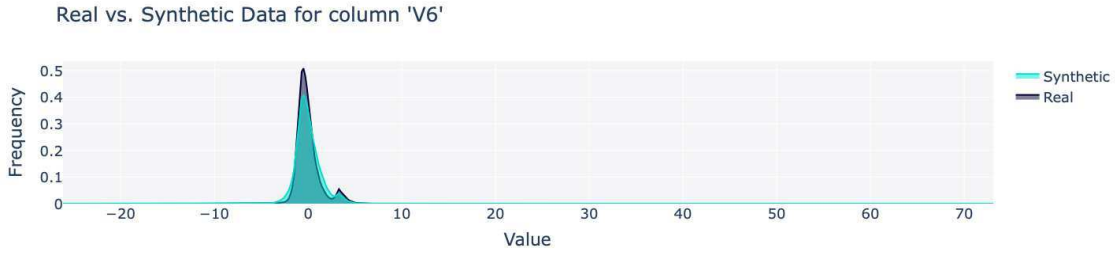

Real vs. Synthetic Data for column 'V1'
Frequency
0.4
0.3
0.2
0.1
0
−50
−40
−30
−20
−10
0
Value
Synthetic
Real


Real vs. Synthetic Data for column 'V2'
Frequency
0.4
0.3
0.2
0.1
0
−70
−60
−50
−40
−30
−20
−10
0
10
20
Value
Synthetic
Real

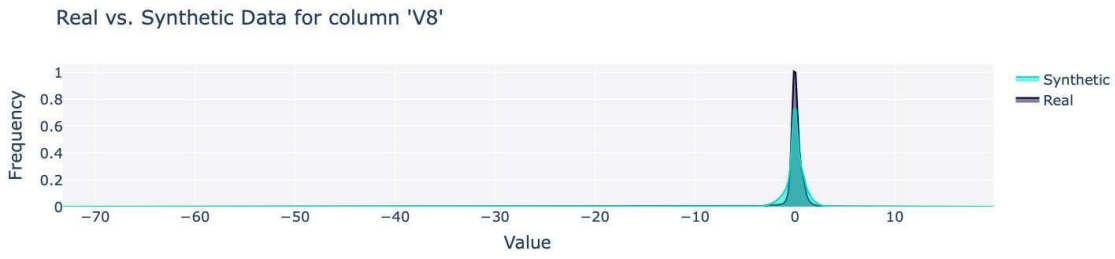

Real vs. Synthetic Data for column 'V3'
Frequency
0.3
0.2
0.1
0
−40
−30
−20
−10
0
Value
Synthetic
Real

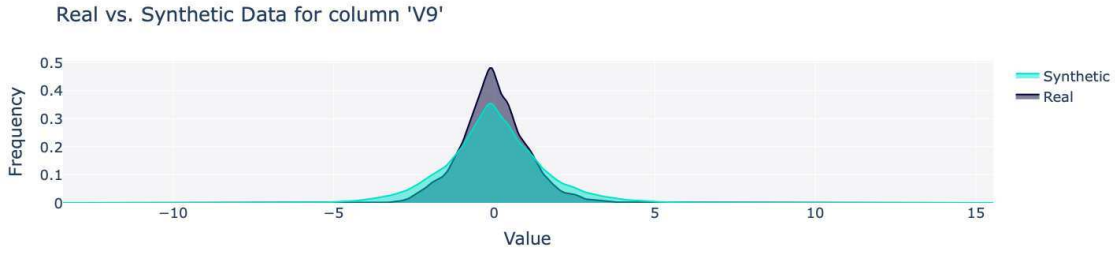

Real vs. Synthetic Data for column 'V4'
Frequency
0.3
0.2
0.1
0
−5
0
5
10
15
Value
Synthetic
Real

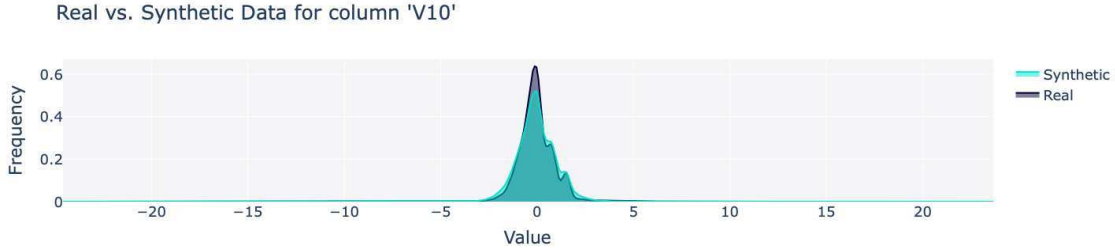

Real vs. Synthetic Data for column 'V5'
Frequency
0.4
0.3
0.2
0.1
0
−100
−80
−60
−40
−20
0
20
Value
Synthetic
Real

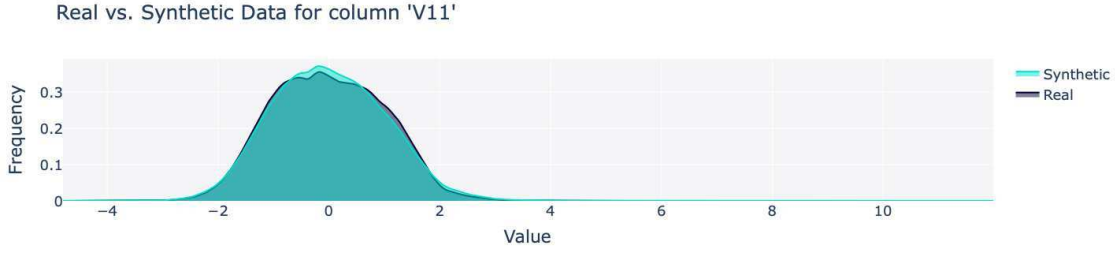

Real vs. Synthetic Data for column 'V6'
Frequency
0.5
0.4
0.3
0.2
0.1
0
−20
−10
0
10
20
30
40
50
60
70
Value
Synthetic
Real


Real vs. Synthetic Data for column 'V7'
Frequency
0.4
0.2
0
−40
−20
0
20
40
60
80
100
120
Value
Synthetic
Real


Real vs. Synthetic Data for column 'V8'
Frequency
Value
Synthetic
Real

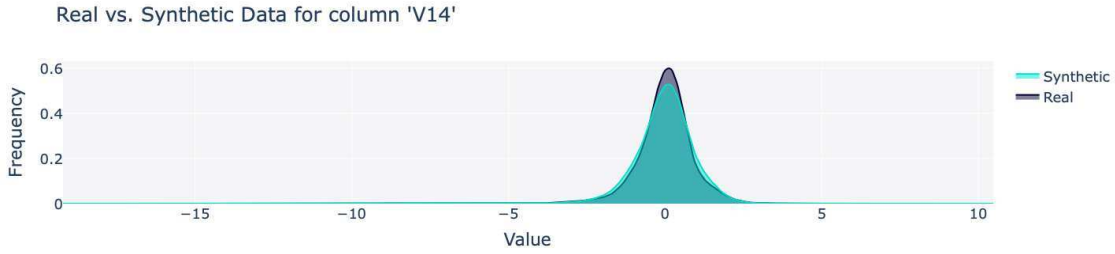

Real vs. Synthetic Data for column 'V9'
Frequency
Value
Synthetic
Real


Real vs. Synthetic Data for column 'V10'
Frequency
Value
Synthetic
Real

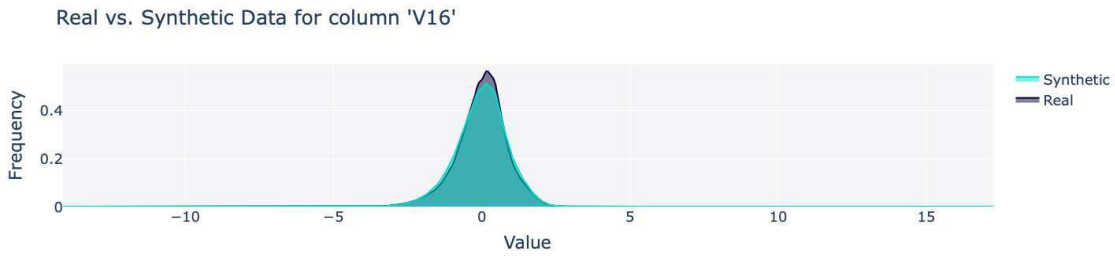

Real vs. Synthetic Data for column 'V11'
Frequency
Value
Synthetic
Real

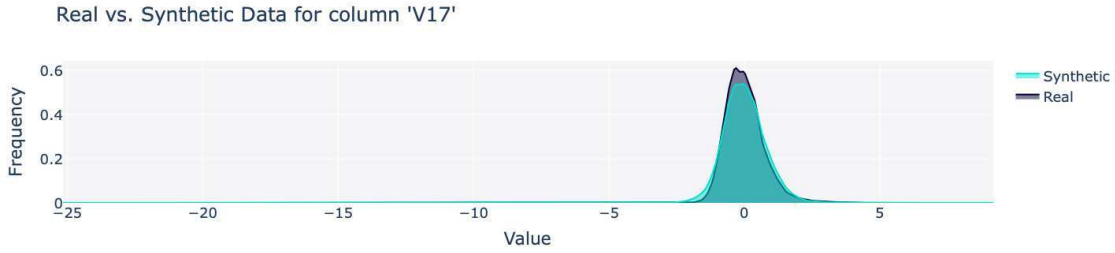

Real vs. Synthetic Data for column 'V12'
Frequency
0.4
0.2
0
−15
−10
−5
0
5
Value
Synthetic
Real


Real vs. Synthetic Data for column 'V13'
Frequency
0.4
0.3
0.2
0.1
0
−4
−2
0
2
4
6
Value
Synthetic
Real

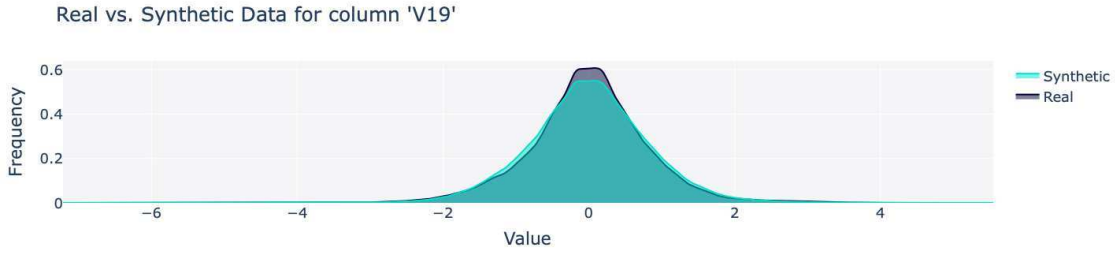

Real vs. Synthetic Data for column 'V14'
Frequency
0.6
0.4
0.2
0
−15
−10
−5
0
5
10
Value
Synthetic
Real

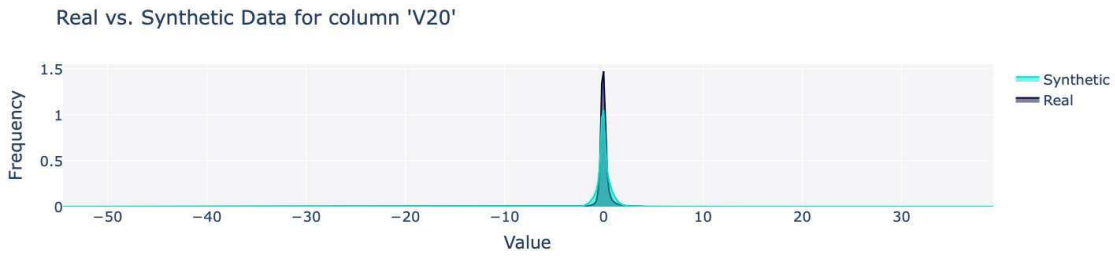

Real vs. Synthetic Data for column 'V15'
Frequency
0.4
0.3
0.2
0.1
0
−4
−2
0
2
4
6
8
Value
Synthetic
Real

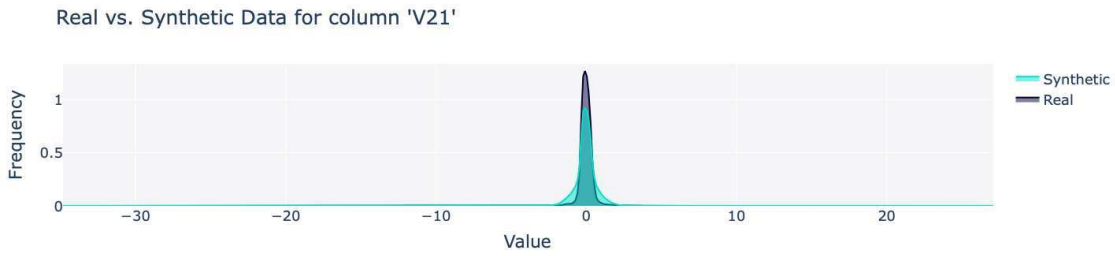

Real vs. Synthetic Data for column 'V16'
Frequency
0.4
0.2
0
−10
−5
0
5
10
15
Value
Synthetic
Real

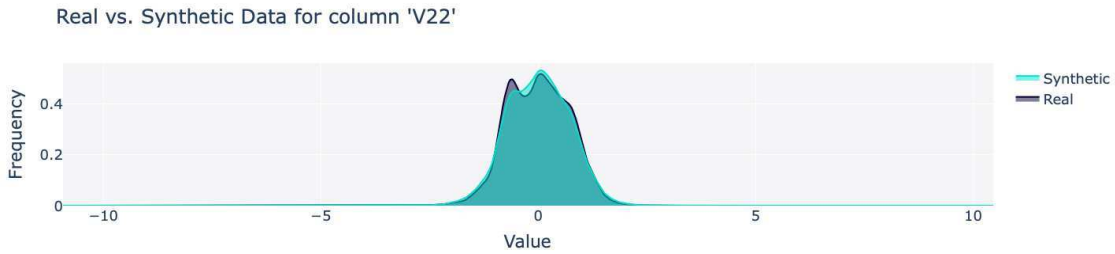

Real vs. Synthetic Data for column 'V17'
Frequency
0.6
0.4
0.2
0
−25
−20
−15
−10
−5
0
5
Value
Synthetic
Real

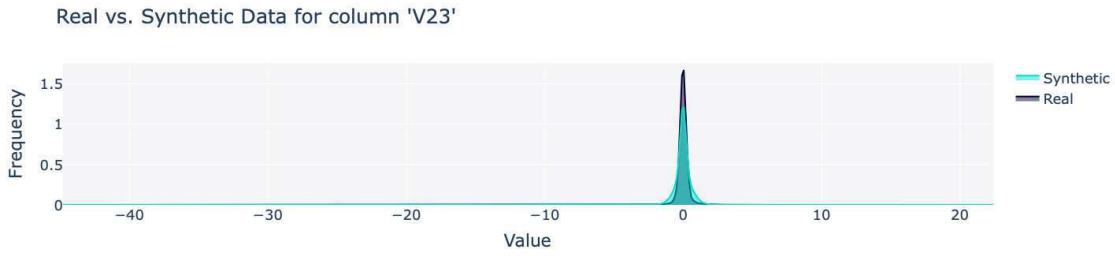

Real vs. Synthetic Data for column 'V18'
Frequency
0.4
0.2
0
−8
−6
−4
−2
0
2
4
Value
Synthetic
Real

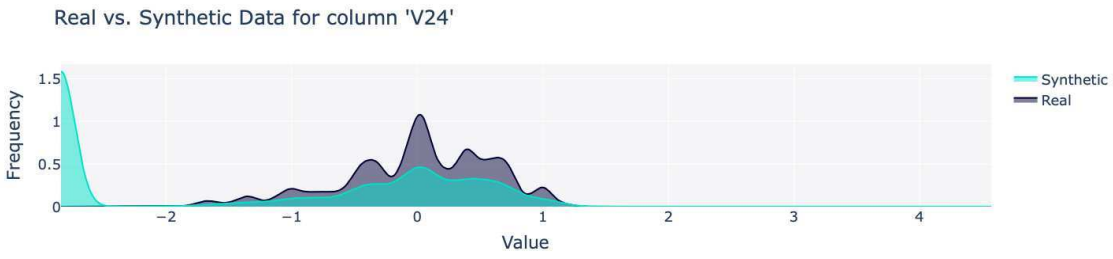

Real vs. Synthetic Data for column 'V19'
Frequency
0.6
0.4
0.2
0
−6
−4
−2
0
2
4
Value
Synthetic
Real

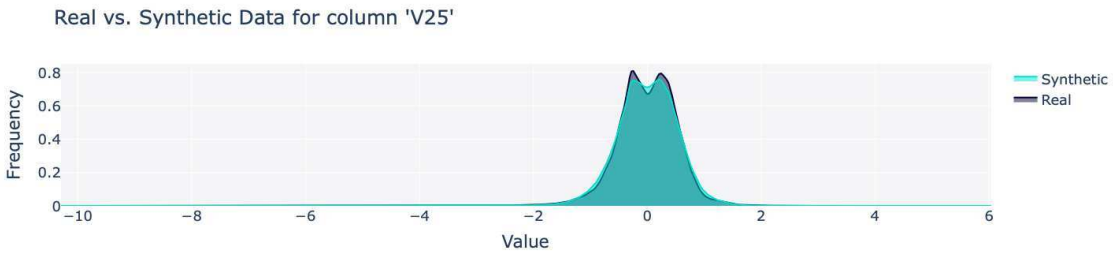

Real vs. Synthetic Data for column 'V20'
Frequency
1.5
1
0.5
0
−50
−40
−30
−20
−10
0
10
20
30
Value
Synthetic
Real


Real vs. Synthetic Data for column 'V21'
Frequency
1
0.5
0
−30
−20
−10
0
10
20
Value
Synthetic
Real

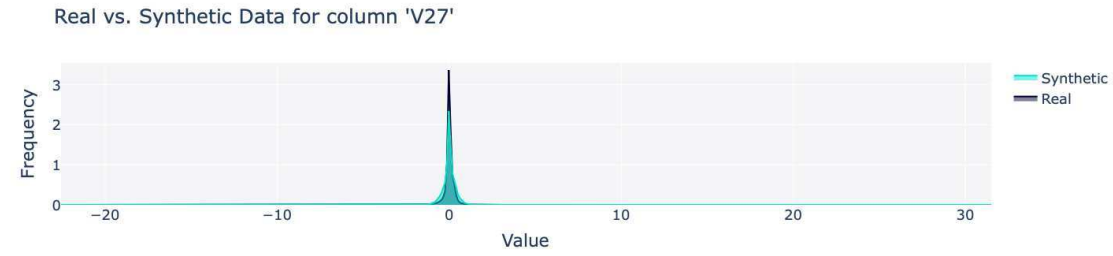

Real vs. Synthetic Data for column 'V22'
Frequency
0.4
0.2
0
−10
−5
0
5
10
Value
Synthetic
Real

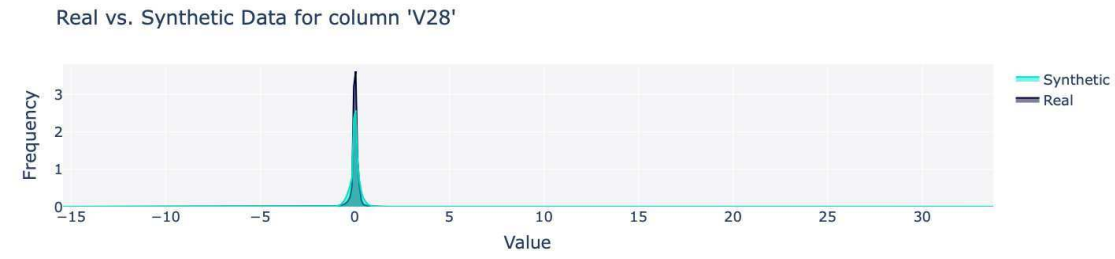

Real vs. Synthetic Data for column 'V23'
Frequency
1.5
1
0.5
0
−40
−30
−20
−10
0
10
20
Value
Synthetic
Real

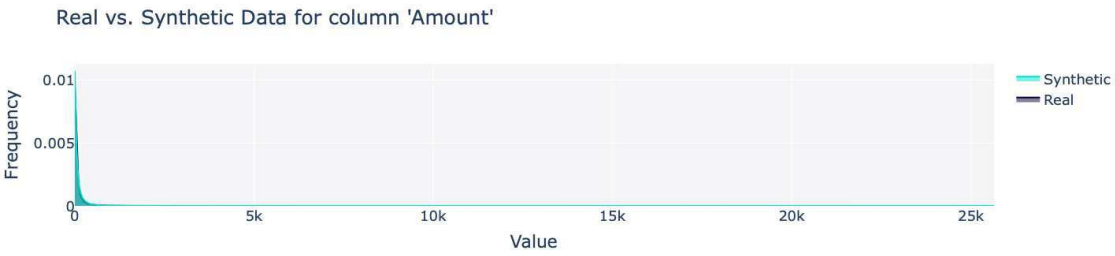

Real vs. Synthetic Data for column 'V24'
Frequency
Value
Synthetic
Real

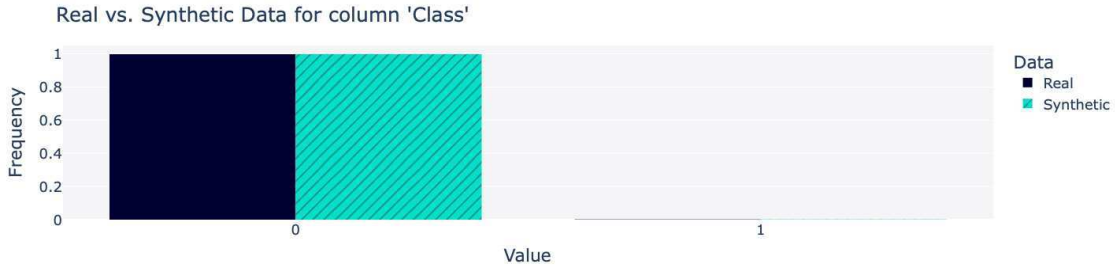

Real vs. Synthetic Data for column 'V25'
Frequency
Value
Synthetic
Real


Real vs. Synthetic Data for column 'V26'
Frequency
Value
Synthetic
Real


Real vs. Synthetic Data for column 'V27'
Frequency
Value
Synthetic
Real







0.4 4. Calculate Performance Scores

[35]:
```
#baseline
X_train_orig = train_data.drop(columns=[target_variable])
y_train_orig = train_data[target_variable].values

#augmented dataset
X_train_aug = synthetic_datasets[0.8].drop(columns=[target_variable])
y_train_aug = synthetic_datasets[0.8][target_variable].values

#holdout set
X_holdout = holdout_set.drop(columns=[target_variable])
y_holdout = holdout_set[target_variable].values
```

```
def train_and_evaluate(X_train, y_train, X_test, y_test):

    model = XGBClassifier(use_label_encoder=False, eval_metric='logloss')
    model.fit(X_train, y_train)

    y_pred = model.predict(X_test)
    y_prob = model.predict_proba(X_test)[:, 1]

    # Basic scores
    acc = accuracy_score(y_test, y_pred)
    prec = precision_score(y_test, y_pred, zero_division=0)
    rec = recall_score(y_test, y_pred, zero_division=0)
    f1 = f1_score(y_test, y_pred, zero_division=0)
    roc_auc = roc_auc_score(y_test, y_prob)

    # Confusion matrix for specificity & sensitivity
    # confusion_matrix returns: [[TN, FP], [FN, TP]]
    tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()

    # Specificity (TNR) = TN / (TN + FP)
    specificity = tn / (tn + fp) if (tn+fp) != 0 else 0.0

    return {
        'accuracy': acc,
        'precision': prec,
        'recall': rec,
        'f1': f1,
        'roc_auc': roc_auc,
        'specificity': specificity,
    }

# Evaluate original dataset
scores_orig = train_and_evaluate(X_train_orig, y_train_orig, X_holdout,␣
 ↪y_holdout)

# Evaluate augmented dataset
scores_aug = train_and_evaluate(X_train_aug, y_train_aug, X_holdout, y_holdout)

##############################################################################
# 3. Generate the Three Categories of Scores


# 1) Single Performance Scores Table
metrics = ["accuracy", "precision", "recall", "f1", "roc_auc", "specificity"]
single_score_dict = {
    "Metric": metrics,
```

```
    "Original Data": [scores_orig[m] for m in metrics],
    "Augmented Data": [scores_aug[m] for m in metrics],
}
single_scores_df = pd.DataFrame(single_score_dict)

# 2) Delta Performance Scores (Aug - Orig)
delta_scores = {}
for m in metrics:
    delta_scores[m] = scores_aug[m] - scores_orig[m]

delta_score_dict = {
    "Metric": metrics,
    "Delta (Aug - Orig)": [delta_scores[m] for m in metrics]
}
delta_scores_df = pd.DataFrame(delta_score_dict)

# 3) Synthesizer Performance Score (sum of all deltas)
synthesizer_performance_score = sum(delta_scores.values())

################################################################################
# 4. Display or Print Results


print("=== Single Performance Scores ===")
print(single_scores_df.to_string(index=False))
print("\n=== Delta Performance Scores ===")
print(delta_scores_df.to_string(index=False))
print("\n=== Synthesizer Performance Score ===")
print(synthesizer_performance_score)
```

```
=== Single Performance Scores ===
     Metric  Original Data  Augmented Data
   accuracy       0.999544        0.999631
  precision       0.918605        0.952941
     recall       0.806122        0.826531
         f1       0.858696        0.885246
    roc_auc       0.974323        0.977259
specificity       0.999877        0.999930

=== Delta Performance Scores ===
     Metric  Delta (Aug - Orig)
   accuracy            0.000088
  precision            0.034337
     recall            0.020408
         f1            0.026550
    roc_auc            0.002936
specificity            0.000053
```

```
=== Synthesizer Performance Score ===
0.08437176670920632
```