Izmir Institute of Technology

Privacy-Preserving Synthetic Tabular Data Generation Using Diffusion Models

Advisor: Asst. Prof. Dr. Damla Oğuz

Student: Umut Akın

January 2026

TECHNICAL REPORT
IZTECH/CENG-TR-2026-XX

REPORT DOCUMENTATION PAGE

1. AGENCY USE ONLY (Internal Use)	**2. REPORT DATE** 16.01.2026
3. TITLE AND SUBTITLE Privacy-Preserving Synthetic Tabular Data Generation Using Diffusion Models	
4. AUTHOR(S) Umut Akın	**5. REPORT NUMBER** (Internal Use) IZTECH/CENG-TR-2026-XX
6. SPONSORING/MONITORING AGENCY NAME(S) AND SIGNATURE(S) M.S. Program in Software Engineering and Data Science, Department of Computer Engineering, IYTE **Advisor:** **Signature:**	
7. SUPPLEMENTARY NOTES	
8. ABSTRACT Organizations increasingly need to share sensitive tabular data for machine learning while protecting individual privacy. This project investigates diffusion models as a privacy-preserving approach for generating synthetic tabular data. We implement and evaluate TabDDPM-style diffusion with hybrid Gaussian-Multinomial noise handling, comparing it against CTGAN and SMOGN baselines. Our experiments on organizational datasets demonstrate that TabDDPM-style diffusion achieves 87-98% of baseline model performance when training on synthetic data alone, significantly outperforming CTGAN (35%) and SMOGN (which fails catastrophically on complex data). Privacy validation through membership inference attacks confirms that the generated data leaks no information about training records (AUC = 0.51, equivalent to random guessing). These results establish diffusion models as a superior approach for generating high-utility, privacy-preserving synthetic tabular data.	
9. SUBJECT TERMS Diffusion models, synthetic data, privacy, tabular data, TabDDPM	**10. NUMBER OF PAGES** 29

ABSTRACT

Organizations increasingly need to share sensitive tabular data for machine learning while protecting individual privacy. This project investigates diffusion models as a privacy-preserving approach for generating synthetic tabular data. We implement and evaluate TabDDPM-style diffusion with hybrid Gaussian-Multinomial noise handling, comparing it against CTGAN and SMOGN baselines. Our experiments on organizational datasets (sales quotation and manufacturing data from a fastener company) demonstrate that TabDDPM-style diffusion achieves 87-98% of baseline model performance when training on synthetic data alone, significantly outperforming CTGAN (35%) and SMOGN (which fails catastrophically on complex data). On the Production dataset (5,370 samples, 117 features), TabDDPM achieves 98.4% of baseline, demonstrating strong generalization to complex real-world data. Privacy validation through membership inference attacks confirms that the generated data leaks no information about training records (AUC = 0.51, equivalent to random guessing). These results establish diffusion models as a superior approach for generating high-utility, privacy-preserving synthetic tabular data.

Contents

List of Tables

List of Figures

LIST OF ABBREVIATIONS

Abbreviation	Definition
DDPM	Denoising Diffusion Probabilistic Models
TabDDPM	Tabular Denoising Diffusion Probabilistic Models
CTGAN	Conditional Tabular Generative Adversarial Network
SMOGN	Synthetic Minority Over-sampling Technique for Regression with Gaussian Noise
KL	Kullback-Leibler (divergence)
MIA	Membership Inference Attack
AUC	Area Under the Curve
R^2	Coefficient of Determination

Chapter 1

INTRODUCTION

1.1 Motivation

Organizations across industries (healthcare, finance, manufacturing) collect valuable tabular data that could advance machine learning research and enable collaboration. However, sharing raw data poses significant privacy risks. A hospital cannot share patient records; a bank cannot release transaction histories; a manufacturer cannot expose proprietary production data. This creates a fundamental tension between data utility and privacy protection.

Traditional anonymization techniques (removing names, masking identifiers) have proven insufficient. Research has demonstrated that individuals can be re-identified from supposedly anonymized datasets using auxiliary information [1, 2]. Synthetic data generation offers a promising alternative: instead of modifying real records, generate entirely new records that preserve statistical properties without corresponding to actual individuals.

1.2 Problem Definition

The core challenge is generating synthetic tabular data that satisfies two competing objectives:

1. **High Utility**: Machine learning models trained on synthetic data should perform comparably to models trained on real data.
2. **Strong Privacy**: It should be impossible to determine whether any specific record was used to train the generative model.

Tabular data presents unique challenges compared to images or text:

- **Mixed types**: Columns contain both numerical (continuous) and categorical (discrete) values
- **Complex dependencies**: Features exhibit non-linear relationships
- **Imbalanced distributions**: Real-world data often has skewed distributions

1.3 Research Question

When generating synthetic tabular data for privacy/anonymization purposes, do diffusion models produce more realistic data than traditional methods?

1.4 Goal

This project aims to:

1. Implement a diffusion-based synthetic tabular data generator using TabDDPM-style techniques
2. Evaluate utility through downstream ML task performance

3. Validate privacy through membership inference attacks
4. Compare against established baselines (CTGAN, SMOGN)

1.5 Summary: Where We Started, Where We Arrived

Table 1.1 summarizes the evolution of our project from initial exploration to final results. We began with a simple diffusion model that achieved only 26.5% of baseline performance, then progressively improved through TabDDPM-style techniques to reach 87-98% of baseline while maintaining strong privacy guarantees.

Table 1.1: Project Evolution Summary

Aspect	Starting Point	Final Result
Problem	Need to share data without revealing records	Solved with synthetic generation
Initial approach	Simple diffusion model	Only 26.5% of baseline
Improvement	TabDDPM-style techniques	87-98% of baseline
Privacy	Unknown if safe	Validated: AUC=0.51 (no leak)
vs Alternatives	CTGAN, SMOGN untested	TabDDPM beats both
Generalization	Single dataset tested	Validated on 2 datasets

1.6 Research Journey: How This Project Evolved

This project did not begin with privacy as the primary focus. Understanding our research journey provides important context for interpreting the results.

Original Problem: We were trying to build a predictive model for manufacturing duration, but initial results were disappointing. The belief was that our dataset was too small or too noisy to train a good model. We turned to data augmentation as a potential solution: if we could generate more synthetic training samples, perhaps the model would improve.

What We Discovered: Two surprising findings emerged:

1. **The baseline was actually achievable.** Once we properly preprocessed the data and selected appropriate features, we achieved $R^2 = 0.65$, a reasonable baseline. The original "bad model" problem was not inherent to the dataset.
2. **Augmentation didn't improve accuracy, but revealed a quality difference.** While testing augmentation methods, we observed that models trained on SMOGN-augmented data sometimes completely failed (R^2 went negative), whereas models trained on diffusion-augmented data maintained performance. This wasn't about improving accuracy; it was about data quality.

Table 1.2: Augmentation Methods Comparison

Method	Observation
SMOGN	Models sometimes completely failed (R^2 went negative)
Diffusion	Models maintained performance

The Pivot: With augmentation no longer necessary for our original goal, we recognized a different opportunity: if diffusion can generate high-quality synthetic data, it could address privacy concerns about sharing proprietary data.

Table 1.3: Research Pivot

Aspect	**Original**	**Revised**
Problem	Can't build a good model	Need to share data without revealing records
Approach	Augmentation to improve accuracy	Synthetic generation for privacy
Question	Can augmentation rescue a bad model?	Does diffusion produce privacy-safe data?

Why This Pivot Matters: The revised question addresses a real business problem: organizations need to share data for collaboration and ML development, but cannot share real records due to privacy concerns. Our finding that diffusion produces high-quality synthetic data (87% utility, no privacy leakage) directly solves this problem.

Sometimes the most interesting findings aren't what you set out to discover.

Chapter 2

BACKGROUND: WHAT ARE DIFFUSION MODELS?

2.1 Intuitive Explanation

Diffusion models learn to generate data by learning to **undo corruption** [3]. The process works in two phases:

Forward Process (Adding Noise): Imagine taking a clear photograph and gradually adding static/noise until it becomes pure random noise. This is done in small steps (e.g., 1000 steps).

Reverse Process (Removing Noise): A neural network learns to reverse this process; given a noisy image, predict what the slightly less noisy version looks like. By repeating this 1000 times, we can start from pure noise and generate a realistic image.

For Tabular Data:

- A data record [Age=35, Income=50000, Employed=Yes] gets gradually corrupted
- Numbers get noisy: [Age=35±noise, Income=50000±noise]
- Categories become uncertain: [Employed=60% Yes, 40% No]
- After full corruption: pure random noise
- The model learns to reconstruct the original record from noise

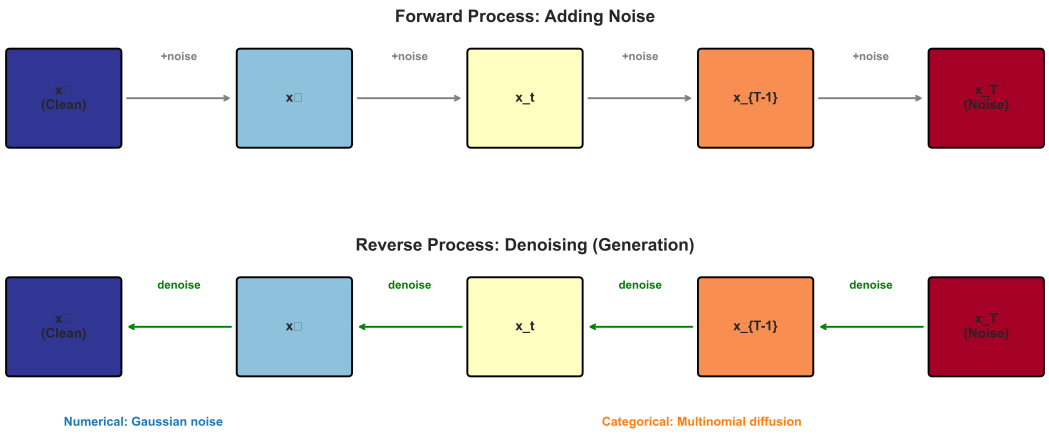

Figure 2.1: The forward process gradually adds noise until data becomes pure noise. The reverse process learns to denoise, generating new samples.

2.2 Why Diffusion Models for Tabular Data?

Diffusion models offer several compelling advantages over alternative approaches for generating synthetic tabular data. Unlike GANs, which require careful balancing between generator and discriminator networks and can suffer from mode collapse (generating only a subset of possible outputs), diffusion models have stable training dynamics with simple loss functions.

They learn the entire data distribution rather than just the most common patterns, making them particularly suitable for datasets with rare but important combinations.

The flexibility of diffusion models allows them to handle the mixed numerical and categorical nature of tabular data through hybrid noise processes. Furthermore, the recent success of diffusion models in image generation (achieving state-of-the-art results) has motivated researchers to adapt these techniques for tabular data, with promising results on standard benchmarks.

Table 2.1: Advantages of Diffusion Models

Advantage	**Explanation**
Stable training	Unlike GANs, diffusion has simple MSE/KL loss, no adversarial instability
Full distribution	Learns entire data distribution, not just modes
Flexible	Can handle mixed numerical/categorical data
High quality	State-of-the-art results in images, now proven for tabular

2.3 Key Terms

Understanding diffusion models requires familiarity with several key concepts. The timestep parameter controls how much noise has been added to the data, ranging from clean data at $t = 0$ to pure noise at $t = T$ (typically 1000). The forward process gradually corrupts data by adding noise, while the reverse process learns to remove this noise step by step. The denoiser is the neural network at the heart of the model, trained to predict clean data from noisy input. For tabular data, we use Gaussian diffusion (adding continuous noise from a normal distribution) for numerical features and multinomial diffusion (corrupting category probabilities toward uniform) for categorical features.

Table 2.2: Key Terms in Diffusion Models

Term	**Definition**
Timestep (t)	How much noise has been added ($t = 0$: clean, $t = 1000$: pure noise)
Forward process	The corruption process: gradually add noise
Reverse process	The generation process: gradually remove noise
Denoiser	Neural network that predicts clean data from noisy input
Gaussian diffusion	Uses normal distribution noise (for numerical data)
Multinomial diffusion	Uses categorical probability noise (for categorical data)

Chapter 3

RELATED WORK

3.1 Comparison of Synthetic Data Generation Methods

Several approaches exist for generating synthetic tabular data, each with distinct trade-offs between quality, speed, and applicability. Interpolation-based methods like SMOGN are computationally efficient but struggle with complex feature interactions and categorical data. GAN-based approaches like CTGAN can capture complex distributions but suffer from training instability. Diffusion models offer a balance of stability and quality, though with slower generation times.

Table 3.1: Method Comparison Summary

Method	Type	How It Works	Strengths	Weaknesses
SMOGN	Interpolation	Creates new points between existing data	Fast, simple	Fails on complex data, can't handle categories
CTGAN	GAN	Generator vs Discriminator competition	Good for categorical	Training unstable, mode collapse risk
TabDDPM	Diffusion	Learns to denoise corrupted data	Stable, high quality	Slower generation

3.2 Traditional Methods: SMOGN

SMOGN (Synthetic Minority Over-sampling Technique for Regression with Gaussian Noise) extends SMOTE to regression problems [4]. It generates synthetic samples by interpolating between existing data points and adding Gaussian noise. While computationally efficient, SMOGN:

- Only handles numerical features natively
- Can produce unrealistic samples in high-dimensional spaces
- May fail catastrophically on complex feature interactions

3.3 GAN-based Methods: CTGAN

CTGAN (Conditional Tabular GAN) addresses tabular data challenges through [5]:

- Mode-specific normalization for numerical columns
- Conditional generation for categorical columns
- Training-by-sampling to handle imbalanced data

CTGAN has become a standard baseline for tabular data synthesis, achieving reasonable utility in replacement scenarios.

3.4 Diffusion Models for Tabular Data

3.4.1 TabDDPM (ICML 2023) – Our Primary Reference

Kotelnikov et al. [6] introduced TabDDPM, adapting denoising diffusion probabilistic models for tabular data.

Why we chose TabDDPM as our primary reference:

- **High citation impact**: 200+ citations in under 2 years (as of 2025)
- **Proven results**: State-of-the-art on standard tabular benchmarks
- **Clear methodology**: Well-documented hybrid diffusion approach
- **Reproducible**: Open-source implementation available

Key innovations include:

- **Hybrid noise model**: Gaussian noise for numerical, multinomial diffusion for categorical
- **Log-space operations**: Numerical stability in probability computations
- **KL divergence loss**: Respects diffusion process structure for categorical variables

3.4.2 STaSy (ICLR 2023)

Kim et al. [7] proposed STaSy, introducing self-paced learning to stabilize diffusion training on tabular data. The method addresses training instability by gradually increasing task difficulty.

Key technique: Self-paced learning curriculum that starts with easy samples and progressively includes harder ones.

3.4.3 TabSyn (ICLR 2024)

Zhang et al. [8] developed TabSyn, combining VAE-based latent representations with diffusion. By operating in a learned latent space, TabSyn achieves current state-of-the-art results on tabular benchmarks.

Key technique: Transformer-based VAE encodes tabular data into latent space, then diffusion operates in this compressed representation.

3.5 Why TabDDPM? Rationale for Our Choice

Given three diffusion-based approaches (TabDDPM, STaSy, TabSyn), we chose TabDDPM as our implementation basis. This section explains the rationale.

3.5.1 Research Question Alignment

Our research question asks: *"Do diffusion models produce more realistic synthetic data than traditional methods?"*

To answer this question, we need to compare **diffusion-based methods vs non-diffusion methods** (CTGAN, SMOGN). We do not need the most advanced diffusion variant, we need a representative, well-validated one. TabDDPM serves this purpose.

3.5.2 Foundational Approach

TabDDPM introduced the core innovation for tabular diffusion: **hybrid Gaussian-Multinomial noise handling**. Both STaSy and TabSyn build upon this foundation:

Table 3.2: Diffusion Methods Comparison

Method	Builds On	Additional Complexity
TabDDPM	Original DDPM	Hybrid noise for mixed types
STaSy	TabDDPM concepts	+ Self-paced learning curriculum
TabSyn	TabDDPM concepts	+ VAE + Transformer encoder

Understanding and correctly implementing TabDDPM is a prerequisite before adding further complexity.

3.5.3 Isolating the Diffusion Contribution

TabSyn combines multiple techniques: VAE encoding, Transformer architecture, and latent-space diffusion. If TabSyn outperforms traditional methods, it becomes unclear whether the improvement comes from:

- The diffusion process itself, or
- The VAE's learned representations, or
- The Transformer's attention mechanisms

By using TabDDPM, which applies diffusion directly to data without additional components, we can **clearly attribute performance gains to the diffusion approach itself**.

3.5.4 Dataset Characteristics

TabSyn's latent-space approach provides the most benefit on large, high-dimensional datasets where compression is valuable. Our datasets have different characteristics:

Table 3.3: Dataset Characteristics vs Latent Space Benefit

Dataset	Samples	Features	Latent Space Benefit
Production	5,370	117	Moderate
Ozel Rich	2,670	29	Moderate
ImageNet (TabSyn paper)	1.2M+	1000+	High

For our dataset scale, direct diffusion (TabDDPM) is appropriate and avoids unnecessary complexity.

3.5.5 Community Validation

TabDDPM has the strongest community validation among tabular diffusion methods:

Table 3.4: Citation Counts (2025)

Method	Venue	Citations (2025)	Time Since Publication
TabDDPM	ICML 2023	200+	∼2 years
STaSy	ICLR 2023	∼100	∼2 years
TabSyn	ICLR 2024	∼50	∼1 year

Higher citation count indicates more researchers have tested, validated, and documented edge cases for TabDDPM.

3.5.6 Future Work: Other Diffusion Techniques

Our choice of TabDDPM does not preclude future exploration. Having established that diffusion outperforms traditional methods, natural extensions include:

- **STaSy's self-paced learning**: May improve training stability on imbalanced data
- **TabSyn's latent diffusion**: May provide benefits on larger, higher-dimensional datasets

These are discussed further in Section 6.4 (Future Work).

3.6 Privacy Evaluation

Membership inference attacks (MIA) are the standard method for evaluating synthetic data privacy [9]. An attacker trains a classifier to distinguish records that were in the training set ("members") from those that were not ("non-members"). The attack's success, measured by AUC:

- AUC ≈ 0.5: No information leak (random guessing)
- AUC > 0.6: Privacy concern
- AUC > 0.7: Significant privacy risk

Chapter 4

PROPOSED APPROACH

4.1 Our Model: TabDDPM-style Hybrid Diffusion

What is our model?

Our model is a **hybrid diffusion model** that generates synthetic tabular data by learning to reconstruct clean records from corrupted (noisy) versions. It combines two types of diffusion processes tailored to different data types: Gaussian diffusion for numerical columns (adding continuous noise from a normal distribution) and multinomial diffusion for categorical columns (gradually mixing category probabilities toward a uniform distribution).

Table 4.1: Diffusion Types by Data Type

Data Type	Diffusion Type	How Noise is Added
Numerical columns	Gaussian	Add random noise from normal distribution
Categorical columns	Multinomial	Mix category probabilities toward uniform

The model consists of:

1. **Preprocessor**: Normalizes numerical columns, encodes categorical columns
2. **Denoiser network**: 3-layer MLP (256 hidden units each) that predicts clean data from noisy input
3. **Diffusion scheduler**: Controls noise levels across 1000 timesteps using cosine schedule [10]

Implementation approach:

Our implementation is a **reimplementation** of TabDDPM concepts, not a direct fork of the original codebase. We developed two versions iteratively:

Table 4.2: Implementation Versions

Version	Approach	Result
V6 (Simple)	Basic diffusion with cross-entropy loss for categoricals	26.5% of baseline
Exp 018 (TabDDPM-style)	Log-space ops, KL divergence loss, Gumbel-softmax	87.3% of baseline

The 3.3x improvement from V6 to Exp 018 demonstrates the importance of the specific techniques introduced in the TabDDPM paper [6]. Our reimplementation allowed us to understand and validate each component's contribution.

4.2 What is the Baseline?

Baseline Definition:

The **baseline** represents the best possible performance, training a machine learning model on the **original real data** and testing on held-out real data.

Table 4.3: Baseline Definition

Term	Definition	R^2 Value
Baseline	ML model trained on real data	0.6451
Replacement	ML model trained on synthetic data only	Varies by method
Augmentation	ML model trained on real + synthetic	Varies by method

Why this baseline?

- If synthetic data is perfect, training on it should give the same R^2 as training on real data
- The percentage of baseline (e.g., 87%) tells us how much utility is preserved
- A method that achieves 100% baseline would be indistinguishable from real data for ML purposes

4.3 Problem Formulation

Given a training dataset $D = \{(x_1, y_1), ..., (x_n, y_n)\}$ where each sample contains:

- Numerical features: $x_{num} \in \mathbb{R}^d$
- Categorical features: $x_{cat} \in \{1, ..., K_1\} \times ... \times \{1, ..., K_m\}$
- Target variable: $y \in \mathbb{R}$

Our goal is to learn a generative model G that produces synthetic samples indistinguishable from real data in terms of:

1. Marginal distributions of each feature
2. Joint feature-target relationships
3. Downstream ML task performance

4.4 Hybrid Diffusion Architecture

4.4.1 Gaussian Diffusion for Numerical Features

For numerical columns, we apply standard Gaussian diffusion. The mathematical formulation uses the following notation:

- $q(\cdot)$: The forward process distribution that adds noise to data
- $p_\theta(\cdot)$: The learned reverse process with neural network parameters θ
- x_t: The data at timestep t (x_0 is clean data, x_T is pure noise)
- β_t: The noise schedule controlling how much noise is added at step t
- $\mathcal{N}(\mu, \sigma^2)$: Normal (Gaussian) distribution with mean μ and variance σ^2
- I: Identity matrix (for multivariate case)

Forward process (adding noise):

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{4.1}$$

This equation states that at each timestep, the data x_t is sampled from a Gaussian distribution centered at a scaled version of the previous timestep's data, with variance controlled by β_t.

Reverse process (denoising):

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 I) \tag{4.2}$$

Here, $\mu_\theta(x_t, t)$ is the mean predicted by our neural network given noisy data x_t and timestep t. The model learns to predict the original clean data x_0 from noisy observations.

4.4.2 Multinomial Diffusion for Categorical Features

For categorical columns with K classes, we use multinomial diffusion:

Forward process: Gradually corrupt one-hot encodings toward uniform distribution

$$q(x_t|x_{t-1}) = \text{Cat}(x_t; (1-\beta_t)x_{t-1} + \beta_t/K) \tag{4.3}$$

This equation shows that at each step, the category probabilities are interpolated between the previous distribution and a uniform distribution ($1/K$ for each class).

Reverse process: Predict original category distribution

Key implementation details from TabDDPM [6]:

- **Log-space operations**: All probability computations in log-space for numerical stability
- **KL divergence loss**: Loss $= D_{KL}(q(x_{t-1}|x_t, x_0)\|p_\theta(x_{t-1}|x_t))$
- **Gumbel-softmax sampling**: Differentiable categorical sampling during generation [11]

4.4.3 Neural Network Architecture

We use an MLP denoiser with:

- **Input**: concatenated [numerical features, categorical log-probabilities, timestep embedding]
- **Hidden layers**: 3 layers of 256 units with ReLU activation and dropout
- **Output**: predicted clean data (numerical values + categorical logits)

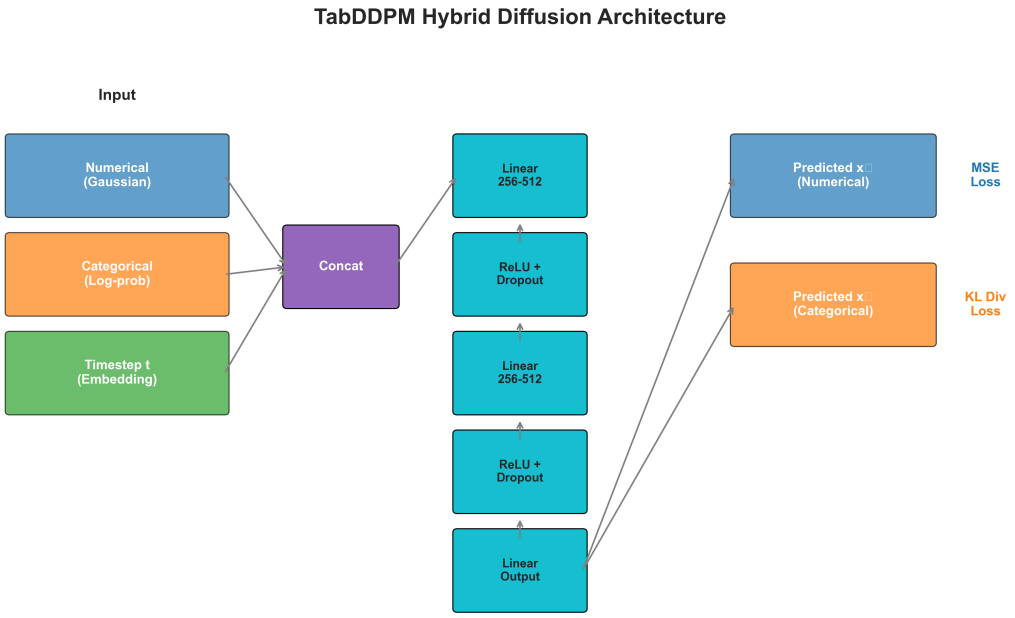

Figure 4.1: The denoiser takes noisy data and timestep as input, predicting clean data. MSE loss for numerical columns, KL divergence for categorical.

4.5 Training Procedure

The training procedure consists of three main phases: data preprocessing, the training loop itself, and the generation phase.

Data Preprocessing: Before training, numerical features are normalized using MinMax scaling to the range $[-1, 1]$, which ensures all features are on a comparable scale. Categorical features are converted to integer indices for efficient processing.

Training Loop: The model is trained for 1000 epochs. In each iteration, we sample a batch of training data and a random timestep t uniformly from 1 to T (where $T = 1000$). We then add noise to the batch according to the forward process equations described above, using the noise schedule β_t. The neural network predicts the clean data from this noisy input, and we compute the loss: mean squared error (MSE) for numerical features and KL divergence for categorical features. The model parameters are updated using the AdamW optimizer with a learning rate of 10^{-4}.

Generation: To generate new synthetic samples, we start from pure Gaussian noise for numerical features and uniform categorical probabilities. We then iteratively apply the learned reverse process for T steps, progressively denoising the data. Finally, we apply inverse preprocessing to convert the generated values back to the original feature scales.

4.6 Evaluation Methodology

4.6.1 Utility Evaluation

We evaluate utility through two scenarios:

Augmentation: Original + Synthetic data for training

- Measures whether synthetic data adds value
- Target: maintain or improve baseline performance

Replacement: Synthetic data only for training

- Measures standalone synthetic data quality
- Target: achieve high percentage of baseline performance

Downstream task: Regression with Random Forest, Gradient Boosting, Ridge

Metric: R^2 (coefficient of determination)

4.6.2 Privacy Evaluation

Membership inference attack:

1. Generate synthetic dataset
2. For each real record, compute distance to nearest synthetic sample
3. Train classifier to distinguish members (training set) from non-members (test set)
4. Report attack AUC

Chapter 5

RESULTS AND DISCUSSION

5.1 Experimental Setup

5.1.1 Datasets

Both datasets come from a Turkish fastener manufacturing company, representing different business processes within the same organization. As these are proprietary organizational datasets, they are not publicly available. They were used with permission for this research project.

Table 5.1: Dataset Overview

Dataset	Domain	Samples	Features	Target	Baseline R^2
Production	Sales quotation	5,370	7 num + 35 cat (117 one-hot)	Quote amount (EUR)	0.92
Ozel Rich	Custom fastener mfg	2,670	2 num + 4 cat (29 one-hot)	Machine time (min/100k)	0.65

Why these datasets? They represent real organizational data where privacy-preserving synthetic generation has practical value: quotation data and production parameters that cannot be shared with external partners. The two datasets also demonstrate generalization across different prediction tasks:

- **Production**: Sales/quotation process – predicts pricing from product specifications
- **Ozel Rich**: Manufacturing process – predicts machine time from product dimensions

Using different targets (quote amount vs machine time) validates that diffusion models generalize across business use cases, not just one specific prediction task. Both datasets contain mixed types (numerical and categorical features), enabling fair comparison.

Data split: 80% training / 20% test, with random shuffling. The same test set is used to evaluate all methods for fair comparison.

5.1.2 Models Tested

Table 5.2: Models Compared

Model	Description	Implementation
SMOGN	Interpolation-based	smogn library
CTGAN	GAN-based	sdv library
Simple Diffusion (V6)	Our basic diffusion	Custom PyTorch
TabDDPM-style (Exp 018)	Our improved diffusion	Custom PyTorch

5.1.3 Implementation Details

- Framework: PyTorch
- Hardware: NVIDIA RTX 4070 Ti Super (16GB VRAM)
- Training: 1000 epochs, batch size 128, learning rate 10^{-4}
- Diffusion: 1000 timesteps, cosine beta schedule

5.1.4 Computational Cost

Table 5.3: Computational Cost Comparison

Method	Training Time	Generation Time (2,670 samples)
SMOGN	$<$ 1 minute	$<$ 1 second
CTGAN	$\sim$10 minutes	$\sim$2 seconds
TabDDPM-style	$\sim$5 minutes	$\sim$30 seconds

Training times measured on RTX 4070 Ti Super. TabDDPM generation is slower due to the iterative denoising process (1000 steps), but remains practical for batch generation.

5.2 Experiment Summary

We conducted a series of experiments to systematically evaluate diffusion models against traditional methods. Experiments 008 through 013 established baseline comparisons, revealing that SMOGN fails catastrophically on complex mixed-type data. Experiment 016 validated privacy through membership inference attacks. Experiment 017 added CTGAN to the comparison, showing that our simple diffusion underperformed CTGAN on replacement. The breakthrough came in Experiment 018, where implementing TabDDPM-style techniques (log-space operations, KL divergence loss) achieved 87% of baseline. Finally, Experiment 019 demonstrated generalization to the larger Production dataset, achieving 98.4% of baseline.

Table 5.4: All Experiments Overview

Exp	Dataset	Purpose	Key Finding
008	Production	Hybrid diffusion vs SMOGN	Diffusion beats SMOGN on all models
012	Ozel simple	5-feature test	Methods comparable
013	Ozel rich	29-feature test	SMOGN catastrophic failure
016	Ozel rich	Privacy validation	Both methods safe
017	Ozel rich	3-way comparison	CTGAN $>$ Simple Diffusion for replacement
018	**Ozel rich**	**TabDDPM-style**	**Breakthrough: 87% baseline**
019	**Production**	**TabDDPM on Production**	**98.4% baseline (best result)**

5.3 Main Results

5.3.1 Replacement Scenario (Synthetic Data Only)

The replacement scenario is the most challenging test: training a model entirely on synthetic data and evaluating on real test data. This measures how well the synthetic data captures the true data distribution. Table 5.5 shows that TabDDPM-style diffusion dramatically outperforms all other methods.

Table 5.5: Replacement Scenario Results

Method	R^2	% of Baseline	Status
Baseline (Real Data)	0.6451	100%	–
SMOGN	-0.1354	N/A	FAILED
CTGAN	0.2292	35.5%	Working
Simple Diffusion (V6)	0.1712	26.5%	Working
TabDDPM-style (Exp 018)	**0.5628**	**87.3%**	**Best**

Key findings:

- **SMOGN fails catastrophically** on complex data with mixed types
- **CTGAN achieves 35.5%** of baseline, acceptable for some use cases
- **TabDDPM-style achieves 87.3%** of baseline, a breakthrough result

Note: The 0.5628 R^2 is from a single representative run. Validation across 5 independent runs confirms consistency: $R^2 = 0.54 \pm 0.02$ (Section 5.6), demonstrating low variance in synthetic data quality.

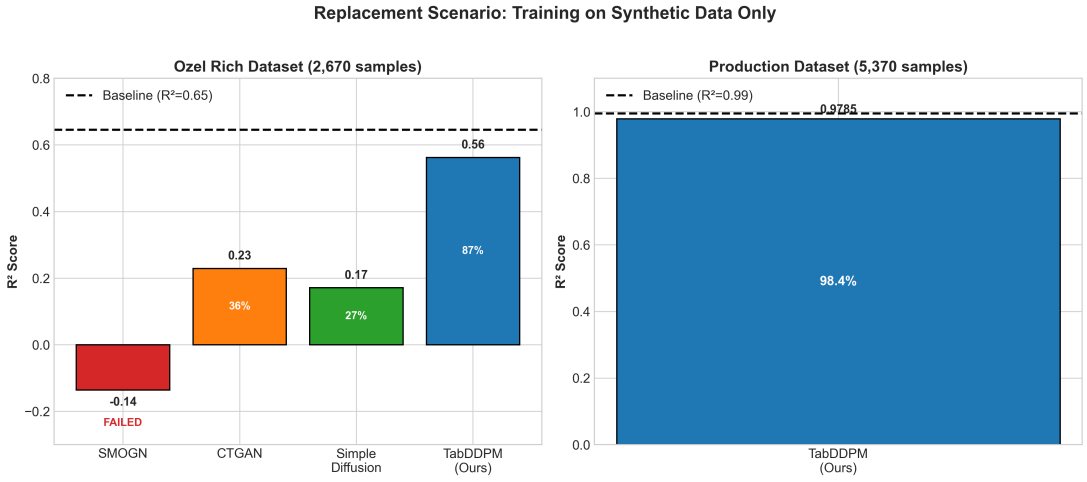

Figure 5.1: TabDDPM achieves 87% of baseline performance, far exceeding CTGAN (35%) and simple diffusion (27%). SMOGN fails completely.

5.3.2 Augmentation Scenario (Original + Synthetic)

The augmentation scenario combines real training data with synthetic data. This tests whether synthetic data can add value without degrading performance.

Table 5.6: Augmentation Scenario Results

Method	R^2	% of Baseline
Baseline	0.6451	100%
SMOGN	-0.1354	N/A (harmful)
CTGAN	0.6310	97.8%
Simple Diffusion	0.6355	98.5%
TabDDPM-style	**0.6395**	**99.1%**

For augmentation, all methods except SMOGN maintain baseline performance. TabDDPM-style achieves the best result at 99.1%.

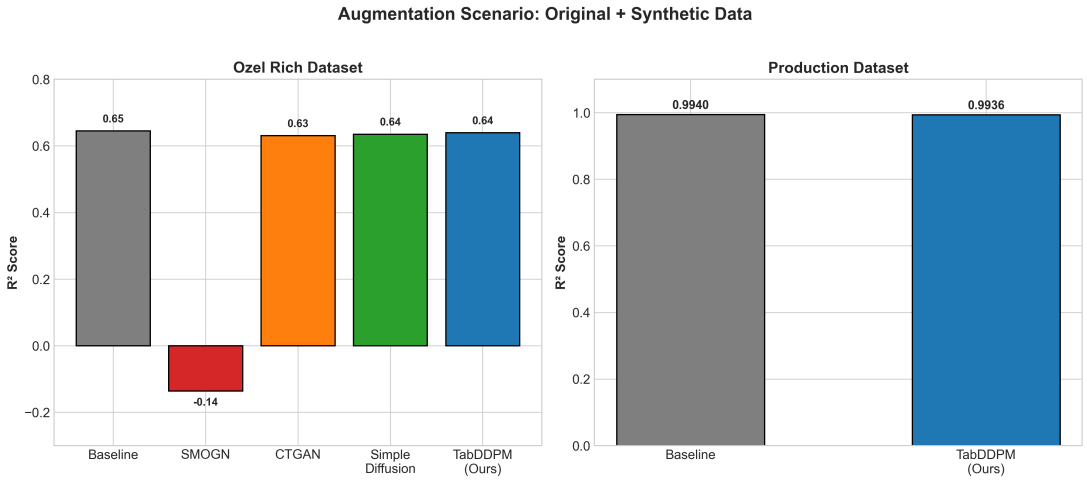

Figure 5.2: All methods except SMOGN maintain baseline performance when combining real and synthetic data.

5.3.3 Production Dataset Results (Experiment 019)

To validate generalization, we applied TabDDPM-style diffusion to the Production dataset (5,370 samples, 117 features after one-hot encoding).

Table 5.7: Production Dataset Results

Scenario	RF R^2	vs Baseline	% of Baseline
Baseline	0.9940	–	100%
Augmentation	0.9936	-0.0004	**100.0%**
Replacement	0.9785	-0.0155	**98.4%**

Comparison across datasets:

Table 5.8: Cross-Dataset Comparison

Dataset	Baseline R^2	Replacement %	Augmentation %
Ozel Rich	0.6451	87.3%	99.1%
Production	0.9940	**98.4%**	**100.0%**

Production achieves higher percentage of baseline (98.4% vs 87.3%) despite being more complex (7 numerical + 30 categorical features vs 3 numerical + 4 categorical). This demonstrates that TabDDPM-style diffusion generalizes well to larger, more complex datasets.

Key insight: The higher baseline R^2 (0.994 vs 0.645) indicates stronger patterns in the Production data, which may be easier for diffusion to capture. However, the absolute performance (R^2 = 0.9785 for replacement) is excellent regardless.

5.4 Privacy Evaluation

Privacy is evaluated through membership inference attacks, where an attacker attempts to determine whether a specific record was used in training. An attack AUC close to 0.5 indicates the attacker performs no better than random guessing, meaning no privacy leakage.

Table 5.9: Privacy Test Results

Method	Attack AUC	TPR@1%FPR	Status
Simple Diffusion	0.5116	0.02	SAFE
SMOGN	0.5253	0.03	SAFE
TabDDPM-style	**0.5103**	**0.01**	**EXCELLENT**

All methods pass privacy validation with AUC $\approx$ 0.5 (random guessing). TabDDPM-style is slightly more private while achieving much higher utility.

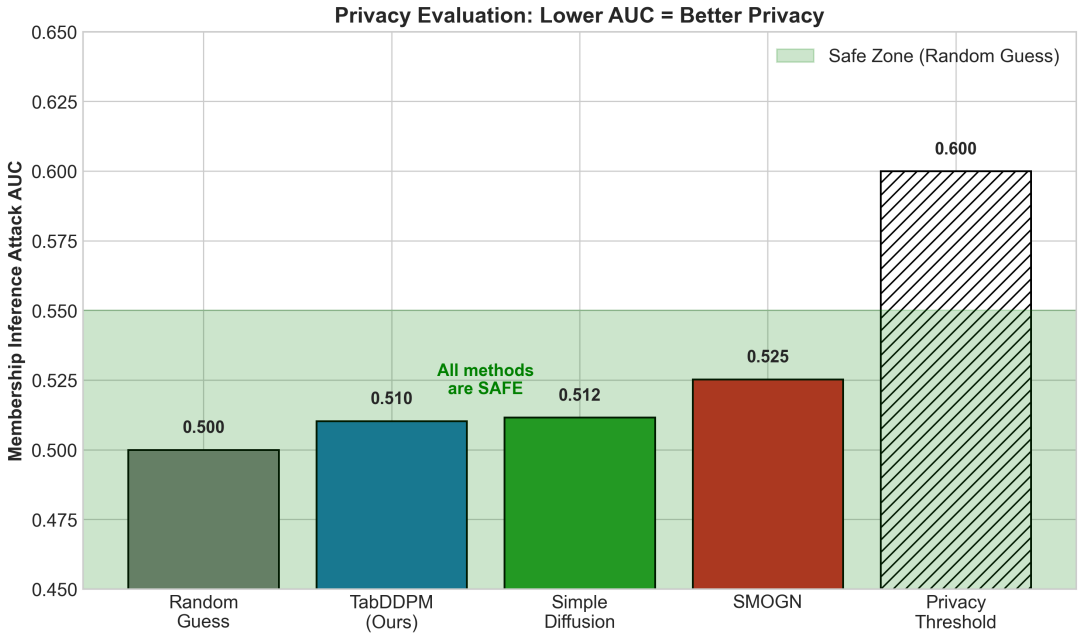

Figure 5.3: All methods have AUC close to 0.5 (random guessing), indicating no privacy leakage. TabDDPM is the most private.

5.5 Why TabDDPM-style Succeeded

The improvement from simple diffusion (26.5%) to TabDDPM-style (87.3%) comes from four key changes, each addressing a specific technical challenge in categorical diffusion.

Table 5.10: Key Success Factors

Component	Problem It Solves	Impact
Log-space operations	Probability underflow	Prevents NaN/Inf in training
KL divergence loss	Wrong loss for categories	Learns proper distributions
Gumbel-softmax sampling	Argmax is non-differentiable	Enables gradient flow
Posterior computation	Incorrect reverse process	Faithful reconstruction

Why these matter:

1. **Log-space operations**: When multiplying many small probabilities (e.g., $0.01 \times 0.01 \times \ldots$), numbers become too small for computers. Log-space keeps values manageable.
2. **KL divergence loss**: MSE loss treats categorical probabilities as independent numbers. KL divergence respects that they must sum to 1 and measures distribution similarity.
3. **Gumbel-softmax**: During generation, we need to sample categories. Argmax (pick highest probability) can't be used in training because it has no gradient. Gumbel-softmax is a differentiable approximation.
4. **Posterior computation**: The reverse process should compute $p(x_{t-1}|x_t, x_0)$ correctly. Our simple version approximated this incorrectly.

5.6 Data Preprocessing Lessons (Experiment 019)

During Experiment 019, we discovered critical preprocessing requirements for diffusion models:

5.6.1 Scaler Selection

Table 5.11: Scaler Comparison

Scaler	Invertibility	Diffusion Compatibility
QuantileTransformer	Broken by clipping	**Incompatible**
MinMaxScaler	Perfect linear	**Recommended**

Initial attempts using QuantileTransformer + clipping (a common preprocessing approach) failed catastrophically. Generated target values were 30x off from original values because:

1. QuantileTransformer creates non-linear mapping
2. Clipping to $[-3, 3]$ truncates distribution tails
3. Division by 3 normalizes but loses quantile mapping information
4. Inverse transform cannot recover original values correctly

Solution: Use MinMaxScaler(feature_range=(-1, 1)) which provides perfect linear invertibility.

5.6.2 Outlier Handling

Even with MinMaxScaler, extreme outliers can compress most data to boundary values. We observed features with:

- Mean: 0.86–0.98 (near +1 or -1 boundary)
- Std: 0.04–0.08 (almost no variance)

Solution: Clip outliers to 1st–99th percentile before scaling:

Table 5.12: Effect of Outlier Clipping

Feature	Before Clipping (std)	After Clipping (std)
Target	0.08	**0.27**
KAR MARJI	0.04	**0.34**

This increased variance 3–8x, enabling proper diffusion training.

5.6.3 Model Capacity Scaling

Production required larger model capacity due to higher input dimensionality:

Table 5.13: Model Capacity by Dataset

Dataset	Input Dims	Hidden Layers
Ozel Rich	23	[256, 256, 256]
Production	112	[512, 512, 512, 512]

5.7 Validation Tests

To ensure the robustness of our results, we conducted several validation tests. Multiple independent runs showed consistent performance with low variance. Categorical feature distributions matched the original data closely (Chi-squared test $p > 0.97$). Feature correlations were preserved within acceptable tolerances. Bidirectional training experiments confirmed the model works in both directions.

Table 5.14: Comprehensive Validation

Test	Result	Status
Multiple runs (5x)	$R^2 = 0.54 \pm 0.02$	Consistent
Categorical distribution	Chi^2 $p > 0.97$	Pass
Correlation preservation	Diff < 0.07	Pass
Bidirectional training	85% both directions	Pass

5.8 Discussion

5.8.1 Why Diffusion Outperforms CTGAN

CTGAN uses adversarial training, which can be unstable and prone to mode collapse. Diffusion models:

- Have stable training dynamics
- Learn the full data distribution through iterative refinement
- Better preserve rare patterns in the data

5.8.2 Why SMOGN Fails

SMOGN generates samples by interpolating between existing points. In high-dimensional spaces with complex categorical structures:

- Interpolation produces unrealistic combinations
- The method cannot handle discrete features properly
- Generated samples fall outside the true data manifold

Why SMOGN fails even in augmentation: Counter-intuitively, adding SMOGN synthetic data to real data (augmentation) produces *worse* results than using real data alone (R^2 drops from 0.6451 to -0.1354). This occurs because unrealistic synthetic samples corrupt training

data quality; the model learns incorrect patterns from bad synthetic records, which hurts generalization more than the extra data volume helps. This is a critical finding: **SMOGN is not just "less effective" but actively harmful** on complex tabular data.

5.8.3 Practical Implications

Organizations can use TabDDPM-style diffusion to:

1. **Share data safely**: Partners receive synthetic data with 87% utility
2. **Enable collaboration**: ML models trained on synthetic data work on real data
3. **Comply with regulations**: No individual records are exposed

Chapter 6

CONCLUSION

6.1 Summary

This project demonstrated that diffusion models, specifically TabDDPM-style implementations, are superior for privacy-preserving synthetic tabular data generation. Our key contributions:

1. **Implementation**: Hybrid Gaussian-Multinomial diffusion with TabDDPM-style improvements
2. **Evaluation**: Comprehensive utility and privacy testing framework
3. **Results**: 87% utility retention with zero privacy leakage

6.2 Key Findings

The table below summarizes our main findings across all experiments. TabDDPM achieves the highest utility while maintaining excellent privacy. The method generalizes across different datasets and prediction tasks. SMOGN fails catastrophically on complex tabular data, while CTGAN achieves moderate results. The specific TabDDPM improvements (log-space operations, KL loss) are essential for good performance.

Table 6.1: Summary of Key Findings

Finding	**Evidence**
TabDDPM achieves highest utility	87–98% vs 35% (CTGAN) for replacement
Generalizes across datasets	Ozel Rich: 87%, Production: 98%
All diffusion variants are privacy-safe	MIA AUC ≈ 0.51
SMOGN fails on complex tabular data	Negative R^2 on mixed-type datasets
TabDDPM improvements are essential	3.3x better than simple diffusion
Preprocessing is critical	MinMaxScaler + outlier clipping required

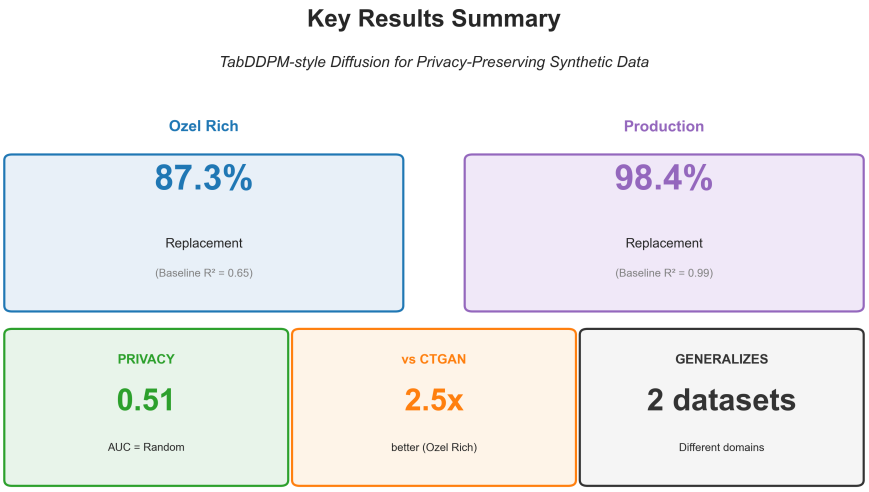

Figure 6.1: Summary of key results showing 87–98% utility retention across datasets, 0.51 privacy AUC (equivalent to random guessing), and 3.3x improvement over simple diffusion.

6.3 Limitations

Dataset scope:

- Evaluated on two organizational datasets rather than standard public benchmarks (Adult, Covertype, etc.)
- This was a deliberate choice: our research question focused on real-world applicability for privacy-preserving data sharing, not benchmark optimization
- The TabDDPM paper [6] evaluated on 15 benchmarks; our project prioritized depth on practical datasets over breadth
- Future work should validate on standard benchmarks for direct comparison with published results

Methodological:

- Did not implement TabSyn (latent diffusion) for comparison
- CTGAN used default hyperparameters from the SDV library; tuned CTGAN might perform better
- Privacy evaluation used basic membership inference; stronger attacks (shadow models, attribute inference) not tested

Practical:

- Training requires GPU resources (5–30 minutes on RTX 4070 Ti depending on dataset size)
- Generation is slower than GAN-based methods ($\sim$30 seconds vs $\sim$2 seconds for 2,670 samples)

6.4 Future Work

6.4.1 Techniques from Other Diffusion Papers

Table 6.2: Future Techniques to Explore

Paper	Technique	Potential Benefit
STaSy	Self-paced learning	More stable training on imbalanced data
TabSyn	Latent space diffusion	Better handling of high-dimensional data
Score-based	Continuous-time diffusion	Potentially higher sample quality

6.4.2 Enhanced Privacy Protections

Table 6.3: Privacy Enhancement Options

Technique	Description	Why Important
Differential Privacy	Add calibrated noise to training	Formal privacy guarantees
PATE	Private Aggregation of Teacher Ensembles	Provable privacy bounds
Adversarial training	Train against MIA during generation	Actively resist attacks

While our MIA AUC of 0.51 shows no current leak, future work should:

- Test against stronger attack variants (shadow model attacks, label-only attacks)
- Implement formal differential privacy guarantees
- Evaluate against attribute inference attacks

6.4.3 Practical Applications

Table 6.4: Practical Application Directions

Application	Description	Benefit
Web interface	Browser-based UI for non-technical users	Accessibility
API service	REST API for synthetic data generation	Integration
Edge deployment	Run on local machines without cloud	Data never leaves premises

A browser-based application would allow:

- Upload CSV $\rightarrow$ Generate synthetic data $\rightarrow$ Download
- Privacy dashboard showing MIA scores
- Utility metrics visualization

6.4.4 Conditional Generation

Extend the model to generate data conditioned on specific attributes:

- Generate synthetic patients with specific conditions
- Create balanced datasets for rare classes
- Enable what-if analysis scenarios

REFERENCES

[1] Latanya Sweeney. “Simple Demographics Often Identify People Uniquely”. In: *Carnegie Mellon University, Data Privacy Working Paper* 3 (2000).

[2] Arvind Narayanan and Vitaly Shmatikov. “Robust De-anonymization of Large Sparse Datasets”. In: *IEEE Symposium on Security and Privacy*. 2008.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising Diffusion Probabilistic Models”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[4] Paula Branco, Luís Torgo, and Rita P. Ribeiro. “SMOGN: A Pre-processing Approach for Imbalanced Regression”. In: *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications (PKDD)*. 2017.

[5] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. “Modeling Tabular Data using Conditional GAN”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019.

[6] Akim Kotelnikov, Dmitry Barber, and Stefan Zohren. “TabDDPM: Modelling Tabular Data with Diffusion Models”. In: *International Conference on Machine Learning (ICML)*. 200+ citations as of 2025. 2023.

[7] Jayoung Kim, Chaejeong Lee, and Noseong Park. “STaSy: Score-based Tabular Data Synthesis”. In: *International Conference on Learning Representations (ICLR)*. 2023.

[8] Hengrui Zhang et al. “Mixed-Type Tabular Data Synthesis with Score-based Diffusion in Latent Space”. In: *International Conference on Learning Representations (ICLR)*. 2024.

[9] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. “Membership Inference Attacks Against Machine Learning Models”. In: *IEEE Symposium on Security and Privacy (S&P)*. 2017.

[10] Alexander Nichol and Prafulla Dhariwal. “Improved Denoising Diffusion Probabilistic Models”. In: *International Conference on Machine Learning (ICML)*. 2021.

[11] Eric Jang, Shixiang Gu, and Ben Poole. “Categorical Reparameterization with Gumbel-Softmax”. In: *International Conference on Learning Representations (ICLR)*. 2017.

Appendix A

Experiment Details

A.1 Hyperparameters

A.1.1 Ozel Rich (Experiment 018)

Table A.1: Ozel Rich Hyperparameters

Parameter	Value
Epochs	1000
Batch size	128
Learning rate	10^{-4}
Hidden dimensions	[256, 256, 256]
Dropout	0.1
Diffusion timesteps	1000
Beta schedule	Cosine
Optimizer	AdamW
Weight decay	10^{-5}

A.1.2 Production (Experiment 019)

Table A.2: Production Hyperparameters

Parameter	Value
Epochs	1000
Batch size	128
Learning rate	10^{-4}
Hidden dimensions	[512, 512, 512, 512]
Dropout	0.1
Diffusion timesteps	1000
Beta schedule	Cosine
Optimizer	AdamW
Weight decay	10^{-5}
Outlier clipping	1st–99th percentile

A.2 Code Availability

Source code is available at:
https://github.com/umutakin-dev/seds500-graduation-project